Genesys Customer Experience Insights Deployment Guide

Genesys Customer Experience Insights 9.0.0

6/24/2024

Table of Contents

Genesys CX Insights 9.x Deployment Guide	**3**
New In This Release	10
Prerequisites: Before you begin installation	**13**
Installing Kubernetes and Docker in online scenarios	20
Installing Kubernetes and Docker in offline scenarios	30
Installing Genesys CX Insights - Kubernetes using descriptors	**48**
Installing Genesys CX Insights - Kubernetes using Helm	**77**
Installing Genesys CX Insights - OpenShift using Helm	**86**
Installing Genesys CX Insights - Docker Compose	**94**
Installing Genesys CX Insights - Podman Compose	118
Installing without Docker	**121**
Installing report editing software	**122**
After Installing Genesys CX Insights	**124**
Accessing CX Insights GUIs	**145**
Uninstalling Genesys CX Insights	**150**
Upgrading Genesys CX Insights	**152**
Best practices when adopting Genesys CX Insights	**157**
Troubleshooting	**164**
Additional resources	**172**

Genesys CX Insights 9.x Deployment Guide

Welcome to the *Genesys CX Insights Deployment Guide*. This document introduces you to the configuration, installation, setup, and start procedures that are relevant to the setup of Genesys Customer Experience Insights (Genesys CX Insights), and the operation of Genesys CX Insights reports. This document is valid only for the 9.0.x releases of Genesys CX Insights, and is intended for deployments of Genesys CX Insights in Genesys Engage on-premises environments.

Important

Should you use the information on this page?

This page applies to Genesys CX Insights **on-premises deployments** only. In such scenarios, always use software that you download from Salesforce, and refer to the on-premises Release Notes.

If you are deploying **Genesys CX Insights in Genesys Multicloud CX private edition**, always use software that you download from the Genesys JFrog repository, and follow the instructions for Genesys Multicloud CX, and refer to the Multicloud CX Release Notes.

Example Report

For versions of this document that have been created for other releases of these products, visit the Genesys Customer Care website, or request the Documentation Library DVD, which you can order by email from Genesys Order Management.

Example Dashboard

About Genesys CX Insights

Genesys CX Insights provides reports and dashboards that summarize contact center activity. Reports display contact center activity using easy-to-read grids, as shown in the figure **Example Report**, while dashboards summarize a wider range of information using a variety of visual devices, such as those shown in the figure **Example Dashboard**. Genesys CX Insights 9.0 is powered by MicroStrategy software. For information about what releases of MicroStrategy software are required, see the Genesys CX Insights Product Alert.

Beginning with release 9.0, Genesys CX Insights replaces Genesys Interactive Insights (GI2), the historical reporting tool used in previous Genesys releases.

> ### Tip
> As with many illustrations in this document, the figures shown here are *thumbnails*; click them to view a larger version.

Choose a deployment type

You can deploy Genesys CX Insights using several methods:

- **Docker Compose** — Docker Compose is typically used for lab or demo environments, where no product traffic exists, or in some cases for small production deployments. A Docker Compose deployment is easier than a Kubernetes deployment because all components are deployed using a single docker-compose file on a single virtual machine (VM). For step-by-step instructions using this simplified deployment method, see: Installing Genesys CX Insights - Docker Compose
- **Kubernetes using descriptors** — Kubernetes deployments are suitable for all production

environments. This method is considerably more complicated than Docker Compose deployments, and deploys Genesys CX Insights across multiple VMs. For step-by-step instructions using this deployment method, start here: Prerequisites: Before you begin installation - Docker and Kubernetes

- **Kubernetes using Helm** — Helm deployments are suitable for all production environments. This method is similar to Kubernetes deployments, in that it uses Kubernetes clusters. However, instead of using Kubernetes descriptors, it uses Helm Charts. For more information, see Deploying GCXI using Helm.
- **OpenShift using Helm** — OpenShift deployments are suitable for all production environments. OpenShift works with Kubernetes to manage containers. For more information, see Deploying GCXI using OpenShift.

Genesys CX Insights Product Alert

Genesys Customer Experience Insights 9.0.0 interoperates with the following releases of other products:

GCXI Release	MicroStrategy Release	Kubernetes/ Docker (or later*)	RAA Release (or later)	Genesys Info Mart Release (or later)	iWD Release (or later)
100.0.035.0000 (9.0.035.00)	MicroStrategy Intelligent Enterprise 11.3.1260 (MicroStrategy 2023 Update 10)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**
100.0.033.0000 (9.0.033.00)	MicroStrategy Intelligent Enterprise 11.3.1060.00468 (MicroStrategy 2023 Update 10)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**
100.0.032.0000 (9.0.032.00)	MicroStrategy Intelligent Enterprise 11.3.0860.01123 (MicroStrategy 2021 Update 8)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**
100.0.031.0000 (9.0.031.00)	MicroStrategy Intelligent Enterprise 11.3.0860.01123 (MicroStrategy 2021 Update 8)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**
100.0.030.0000 (9.0.030.00)	MicroStrategy Intelligent Enterprise 11.3.0760.00770 (MicroStrategy 2021 Update 7)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**

GCXI Release	MicroStrategy Release	Kubernetes/ Docker (or later*)	RAA Release (or later)	Genesys Info Mart Release (or later)	iWD Release (or later)
100.0.029.0000 (9.0.029.00)	MicroStrategy Intelligent Enterprise 11.3.0560.0065 (MicroStrategy 2021 Update 5.1) / MicroStrategy Workstation 11.3.630.694 (Framework Build 11.3.63)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**
100.0.028.0000 (9.0.028.00)	11.3.0560.0065 (MicroStrategy 2021 Update 5.1)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**
100.0.027.0001 (9.0.027.01)	11.3.0460.00602 (MicroStrategy 2021 Update 4.1)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.011.01	8.5.014.34	**
100.0.026.0001 (9.0.026.01)	11.3.0460.00602 (MicroStrategy 2021 Update 4.1)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
Note: In deployments that include Genesys Info Mart 8.5.116.26 or later, the RAA.log file is not generated.					
100.0.026.0000 (9.0.026.00)	11.3.0460.00602 (MicroStrategy 2021 Update 4.1)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
100.0.025.0001 (9.0.025.01)	11.3.0300.11047 (MicroStrategy 2021 Update 3)	v1.20..v.1.23 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
100.0.024.0000 (9.0.024.00)	11.3.0300.11047 (MicroStrategy 2021 Update 3)	v1.20..v.1.22 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
100.0.023.0001 (9.0.023.01)	11.3.0200.19374 (MicroStrategy 2021 Update 2)	v1.19..v1.21 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
100.0.023.0000 (9.0.023.00)	11.3.0200.19374 (MicroStrategy 2021 Update 2)	v1.19..v1.21 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
100.0.021.0000 (9.0.021.00)	11.3.0200.19374 (MicroStrategy 2021 Update 2)	v1.18...v1.20 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
100.0.020.0000 (9.0.020.00)	11.3.0100.18093 (MicroStrategy 2021 Update 1)	v1.18...v1.20 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**

GCXI Release	MicroStrategy Release	Kubernetes/ Docker (or later*)	RAA Release (or later)	Genesys Info Mart Release (or later)	iWD Release (or later)
9.0.019.01	11.3.0100.18093 (MicroStrategy 2021 Update 1)	v1.18...v1.20 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
9.0.019.00	11.3.0000.16816 (MicroStrategy 2021)	v1.18...v1.20 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
9.0.018.00	11.3.0000.16816 (MicroStrategy 2021)	v1.17...v1.19 / 18.06-ce...19.03-ce	9.0.001.10	8.5.014.34	**
9.0.017.01	11.2.0300.40207 (MicroStrategy 2020 Update 3)	v1.17...v1.19 / 18.06-ce...19.03-ce	9.0.001.07	8.5.014.34	**
9.0.016.03	11.2.0300.40207 (MicroStrategy 2020 Update 3)	v1.16...v1.19 / 18.06-ce...19.03-ce	9.0.001.03	8.5.014.34	**
** In scenarios where Genesys CX Insights release 9.0.016.03 or later is deployed with iWD release 9.0.012.07 or later, or with iWD 8.5.108.14 or a later 8.5 release, Genesys CX Insights automatically enables only those prompts, metrics, and attributes in the Genesys CX Insights for iWD project that are supported by the installed release of iWD.					
9.0.016.02	11.2.0300.40207 (MicroStrategy 2020 Update 3)	v1.16...v1.19 / 18.06-ce...19.03-ce	9.0.001.03	8.5.014.34	9.0.012.07
9.0.015.04	11.2.0.0200.39920 (Microstrategy 2020 Update 2)	v1.15...v1.18 / 18.06-ce...19.03-ce	8.5.011.04	8.5.014.34	8.5.108.14 / 9.0.012.07
9.0.015.02	11.2.0.0200.39920 (Microstrategy 2020 Update 2)	v1.15...v1.18 / 18.06-ce...19.03-ce	8.5.011.04	8.5.014.34	8.5.108.14 / 9.0.012.07
9.0.015.01	11.2.0100.38862 (Microstrategy 2020 Update 1)	v1.15.0...v1.18.1 / 18.06.1-ce...19.03.8-ce	8.5.011.04	8.5.014.34	8.5.108.14 / 9.0.012.07
9.0.014.03	11.2.0000.38225 (Microstrategy 2020)	v1.15.0...v1.18.1 / 18.06.1-ce...19.03.8-ce	8.5.011.03	8.5.014.34	8.5.108.14 / 9.0.012.07
9.0.014.02	11.2.0000.38225 (Microstrategy 2020)	v1.15.0-1.17.3 / 18.06.1-ce...19.03.5-ce	8.5.011.03	8.5.014.34	8.5.108.14 / 9.0.012.07
9.0.013.01	11.1.0000.0123 (MicroStrategy 2019)	v1.15.0-1.17.0 / 18.06.1-ce...19.03.5-ce	8.5.011.02	8.5.014.26	8.5.108.11 / 9.0.011.09
9.0.012.01	11.1.0000.0123 (MicroStrategy 2019)	v1.14.1-1.16.2 / 18.06.1-ce...18.09.1-ce	8.5.010.01	8.5.013.06 (8.5.014.09 for Chat Thread reporting) (8.5.014.19 for Genesys Predictive	8.5.108.03 / 9.0.008.07

GCXI Release	MicroStrategy Release	Kubernetes/ Docker (or later*)	RAA Release (or later)	Genesys Info Mart Release (or later)	iWD Release (or later)
				Routing reporting)	
9.0.011.03	11.1.0000.0123 (MicroStrategy 2019)	v1.14.1-v1.14.3 / 18.06.1-ce...18.09.1-ce	8.5.009.04	8.5.013.06 (8.5.014.05 for Chat Thread reporting)	8.5.108.03 / 9.0.008.07
9.0.011.02	11.1.0000.0123 (MicroStrategy 2019)	v1.14.1-v1.14.3 / 18.06.1-ce...18.09.1-ce	8.5.009.04	8.5.013.06 (8.5.014.05 for Chat Thread reporting)	8.5.108.03 / 9.0.008.07
9.0.011.00	11.1.0000.0123 (MicroStrategy 2019)	v1.14.1-v1.14.3 / 18.06.1-ce...18.09.1-ce	8.5.009.04	8.5.013.06 (8.5.014.05 for Chat Thread reporting)	8.5.108.03 / 9.0.008.07
9.0.010.05	11.1.0000.0123 (MicroStrategy 2019)	v1.13.1...1.13.4 / 18.06.1-ce...18.09.1-ce v1.12.2/18.06.1-ce	8.5.008.00	8.5.011.18 (8.5.013 for Media Neutral reporting)	8.5.108.03 / 9.0.008.07
9.0.010.04	11.1.0000.0123 (MicroStrategy 2019)	v1.13.1...1.13.4 / 18.06.1-ce...18.09.1-ce v1.12.2/18.06.1-ce	8.5.008.00	8.5.011.18 (8.5.013 for Media Neutral reporting)	8.5.108.03 / 9.0.008.07
9.0.009.00	10.11.0100.0011	v1.13.1/ 18.09.1-ce v1.12.2/18.06.1-ce	8.5.007.00	8.5.011.15 (except Co-browse)	na
9.0.008.00	10.11.0100.0011	v1.12.2/ 18.06.1-ce v1.11.3/18.06.1-ce (docker 1.12.3 for demo)	8.5.006.00	8.5.011.15	na
9.0.007.03	10.11.0100.0011	v1.11.3/ 18.06.1-ce v1.10.5/17.03.1-ce (docker 1.12.3 for demo)	8.5.005.03 (8.5.005.03 to support latest Callback functionality)	8.5.01* (8.5.010.16 to support latest Callback functionality)	na

* GCXI is tested with the Docker/Kubernetes releases specified in the table, and is compatible with subsequent releases that are guaranteed to be backward compatible with the stated release. For more information about what Docker releases interwork with a given Kubernetes release, see the Kubernetes release notes.

Important

Genesys CX Insights supports reporting on asynchronous chat (Async chat). Note, however, that this feature requires a specific release of Genesys Info Mart. Check with your Genesys representative to see if a release of Genesys Info Mart with Async chat support is available.

- Interaction Concentrator (ICON) 8.*x* and Genesys Info Mart 8.*x* might report ACW even if the agent does not accept a call (or consultation) but has ACW unrelated to the call. This recording leads to incorrect ACW, Handle Time, and related measures in some of the aggregates and reports. Refer to the release notes of these products for further information. (ER 258562765)
- The Genesys CX Insights reports count an invitation for collaboration that an agent declines as **Consult Received Accepted** when the agent uses Genesys Agent Desktop (or a custom desktop using the same SDK) to decline the invitation. Refer to ER 247946331 in the *Genesys Info Mart 8.0 Release Notes* for additional information. (ERs 250850268, 247946331)
- Because Interaction Concentrator 8.*x* does not support one-step conferences for SIP Server, Genesys Info Mart cannot populate the proper conference-related fields in the INTERACTION_RESOURCE_FACT table. As a result, this activity is not reported within the Genesys CX Insights reports. (ER 239356191)
- Genesys CX Insights is powered by Microstrategy software. Additional documentation for Microstrategy software is available at www.microstrategy.com

New In This Release

This section describes the changes that have been incorporated within this guide since the 9.0.0 release of Genesys CX Insights.

Genesys CX Insights 9.0.019

- Added a procedure describing how to upgrade the meta and history database.

Genesys CX Insights 9.0.016

- New deployment options are now supported:
  - OpenShift deployment
  - Helm Deployment

Genesys CX Insights 9.0.015

- Various deployment procedures are updated to use NGINX 0.30.0.
- Older procedures and steps that are no longer required in this release are removed, throughout the document.
- Renamed several pages and headings for added clarity.

Genesys CX Insights 9.0.014

- Minor updated to Installing Genesys CX Insights instructions for clarity, and removed UTF_OPTION.
- Updated the Deploying Kubernetes clusters procedures to reflect new sysctl / IPC requirements.
- Updated the Configure these RAA options to automatically enable GCXI features table.
- Extensive updates to deployment steps and other information in this document.

Genesys CX Insights 9.0.013

- Updates on After Installing Genesys CX Insights:

  - The default user accounts (Developer, Editor, Viewer), are now disabled by default. A new container management variable can re-enable the default accounts. See Users and Groups.
  - Added information about hiding unsupported or unwanted reports.
  - Added information about base URL redirect.
- Added information about HA Architecture including a figure.
- Updated NGIX procedure to remove **nginx-configmap.yaml** and **nginx-daemon.yaml**.

Genesys CX Insights 9.0.012

- Removed information about default passwords.
- Removed information about the Genesys Docker Repository.

Genesys CX Insights 9.0.011

- Updated the Installation procedure to use a later Ingress controller.

Genesys CX Insights 9.0.010

This release includes several corrections and enhancements to deployment information and procedures, including:

- A new deployment procedure is provided for environments without online access: Installing Kubernetes and Docker in offline scenarios. This change also includes a new page, Before you install GCXI and updates, including renaming, to the existing page Installing Kubernetes and Docker in online scenarios.
- Added information about configuring Custom Data Access Restrictions
- Changes relating to multiple-project support:
  - Changes to the gcxi.properties
  - Changes to user groups, described in View the project
- Added information about automatic configuration — Some Genesys CX Insights reporting features and the associated objects (including certain folders and reports) are not needed in all deployments, or may require additional configuration steps. The Genesys CX Insights deployment routine now automatically enables these reporting features based on the features you enable in RAA. For more information, see Pre-installation configuration.
- Miscellaneuous updates and corrections:
  - Updates to IP file names
  - Added additional information about License keys

Genesys CX Insights 9.0.009

This release includes minor corrections and enhancements to deployment procedures, including changes or corrections to file names, paths, and passwords.

Genesys CX Insights 9.0.007

This is the initial release of Genesys Customer Experience Insights (CX Insights).

Other Changes

For information about other changes since the initial release, refer to the *New in 9.0.0* and *9.0 Product Alerts* links on the Genesys CX Insights page.

Prerequisites: Before you begin installation

There are several supported deployment methods for Genesys CX Insights:

- **Kubernetes using descriptors** — Prerequisites for this deployment type are described in detail on this page. Complete the steps on this page as required in your environment, and then proceed to deploy GCXI as described later on this page.
- **Kubernetes using Helm** — Prerequisites for this deployment type are described in detail on this page. Complete the steps on this page as required in your environment, and then see Deploying GCXI using Helm.
- **OpenShift using Helm** — Prerequisites and information about how to install OpenShift, are provided on the Red Hat OpenShift site. For GCXI deployment instructions, see Deploying GCXI using OpenShift.
- **Docker Compose** — Installing Genesys CX Insights - Docker Compose, which is suitable for testing or development, or for very small production environments. Prerequisites for this deployment method are described on the Installing Genesys CX Insights - Docker Compose page, which also describes deployment steps.

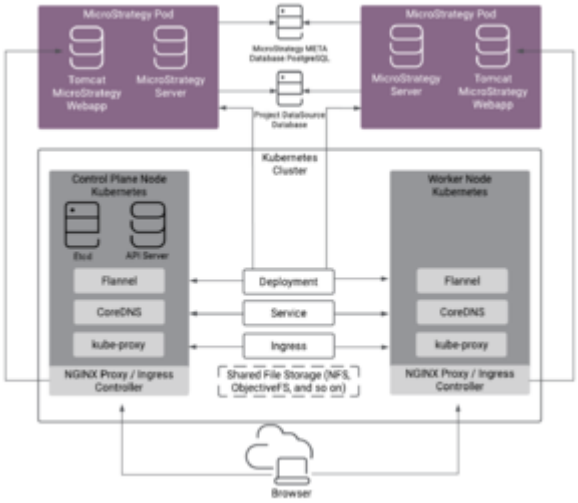
This page describes prerequisites that must be met before you can install Genesys Customer Experience Insights (Genesys CX Insights) in a Kubernetes production environment. For example, you must prepare a suitable Linux server environment, and install Kubernetes, you must plan how you will handle the meta database, and you must identify the compatible releases of the software you will need.

> **Important**
>
> The MicroStrategy server instance that runs in the container includes a temporary pre-activated license key, which is required for the operation of MicroStrategy. Request a replacement key from your customer care representative; you will install the new key during the deployment process.

1. Ensure that your system meets minimum hardware and system requirements

- **Required number of machines** — These could be real machines, virtual machines, or cloud machines such as EC2 instances in AWS, and must be configured as required to support the indicated Linux version. Ensure, for example, that IPV6 is enabled.
  - **In non-High-Availability (HA) deployments**, at least two machines (nodes), each with a supported version of Linux with the **systemd** suite installed. Genesys recommends Red Hat Enterprise Linux 7.5 (or a later 7.x release) / CentOS Linux 7.5 (or a later 7.x release). Other clones of Red Hat Enterprise Linux 7, such as Oracle Linux 7.5, should also work correctly.

- In a two-machine configuration, the Control plane node hosts mstr-01, and a worker node hosts PostgreSQL and mstr-02.
- Optionally, add a third machine if you prefer to deploy each mstr service on a separate node.



High -Availability (HA) Deployment Architecture

- **In High-Availability (HA) deployments**, the following minimums apply:
  - Small/medium customers — Two nodes for MicroStrategy, plus one node for custom cluster PostgreSQL installation.
  - Large customer — Two nodes with MicroStrategy, plus external custom cluster PostgreSQL installation.

    In HA deployments with Kubernetes, Genesys recommends:

  - A 3-node Control plane cluster, which can be either:
    - Three tiny Control plane Kubernetes nodes (meeting minimum Kubernetes requirements), plus two worker nodes (meeting Genesys CX Insights minimum requirements).

      OR

    - One tiny node for standalone Kubernetes Control plane node, plus two worker nodes that also host Kubernetes Control plane nodes.

      See About HA deployments for more information.
- In all scenarios:
  - Identify one machine that you designate the mstr-01; other machine(s) are designated as mstr-02, mstr-03, and so on.
  - Ensure that you have access to an account with root access.
- **Specifications of each machine** — In general, you can deploy MicroStrategy and Genesys CX Insights on any Linux platform with appropriate resources to deploy and run Kubernetes and Docker. However, MicroStrategy / Genesys CX Insights can require significant resources, so note the following minimums for each machine:
  - 64-bit compatible CPU architecture. (2 or more CPUs).
  - 10 GB for each GCXI container, and 2 GB for the PostgreSQL container. If your deployment scenario

involves more than one container on a machine, then the memory requirements/recommendations increase accordingly; for example, if you were to deploy the GCXI container and PostGreSQL on a single host, a minimum of 10 GB + 2 GB = 12 GB is required. Deployment of a GCXI container on a machine with less than 10 GB is not recommended, and requires changes in the `gcxi.yaml` file. Production deployments commonly reserve 16 – 64 GB RAM for each container.

- 40 GB of available disk space if loading images from a repository, 80 GB if loading from a local drive.
  Some deployments use a separate machine for PostgreSQL and gcxi_control, which requires a minimum configuration of 2GB RAM + 1 CPU. The recommended configuration varies depending on the load in your environment, but generally is 4 GB + 2 CPUs. This reflects the requirements for PostGreSQL, as gcxi_control doesn't demand significant resources.

For information about requirements to install Kubernetes, see Installing kubeadm.

About HA deployments

Because a Genesys CX Insights environment consists of several interoperating components, there are several levels of HA configuration that you can optionally deploy. Each type of HA configuration is independant from the others -- you can configure any, all, or none of the following components as HA:

- **GCXI HA** — In a Gensys CX Insights HA deployment, MicroStrategy clusters are used to distribute workers (at least 2, but not more than 8) across multiple machines. The MicroStrategy containers in an HA cluster use an *active-active* model, so if any container fails, Genesys CX Insights continues to operate normally.
- **Kubernetes platform HA** — In a Kubernetes HA deployment, redundant infrastructure is used to run the Kubernetes management infrastructure on several nodes, configured in such a way that the failure of any single node does not interfere with normal Kubernetes operation. Each redundant node stores all Kubernetes state information (etcd). Genesys CX Insights requires no configuration changes to work with HA Kubernetes.
  Kubernetes supports a wide variety of HA options, the simplest consisting of three Kubernetes Control plane nodes. For more information about specific HA deployment options, see Kubernetes HA topology. Note that each MicroStrategy instance that is part of an HA cluster must use a unique path to store log files. The creation of MicroStrategy clusters can fail in scenarios where more than one MicroStrategy instance tries to write logs into a common folder, such as **/mnt/log**.
- **PostgreSQL HA** — PostgreSQL is used for GCXI server metadata; in an HA PostgreSQL deployment, Genesys CX Insights can continue to operate when PostgreSQL container fails. Standard GCXI deployments use a single-node PostgreSQL deployment, and this documentation describes only single node (non-HA) PostgreSQL. To configure HA for the server metadata, follow PostgreSQL documentation, and use standard PostgreSQL database capabilities and administration procedures to configure an external database, and point Genesys CX Insights Kubernetes to it. Genesys CX Insights requires no configuration changes to work with HA PostgreSQL.

2. Plan to accommodate the meta database

Depending on whether you choose to deploy an external PostgreSQL server for the *meta* database, one of the following statements applies:

- If you use an external PostgreSQL server to store the MicroStrategy meta database, ensure that your

PostgreSQL server is configured with a compatible release:

- For new installations of release 100.0.020 and later, PostgreSQL 12 is required for the meta database. If you upgrade an existing deployment to release 100.0.020 or later, it will continue to work with existing PostreSQL releases.

- If you wish to avoid deploying and managing a PostgreSQL server, use the prepackaged PostgreSQL server provided in the Installation Package. For more information about this option, see Installing Genesys CX Insights in production environments.

3. Identify compatible software releases

Genesys CX Insights requires that your environment contain supported releases of the following components.

- Genesys recommends that you use the latest supported version wherever possible.
- See Genesys CX Insights Product Alert for detailed information about supported releases.

4. Pre-installation configuration

Complete the following configuration changes, referring to the operating system documentation for more information if needed:

1. Configure shared memory settings — MicroStrategy requires that you preconfigure shared-memory settings on the host operating system. See the MicroStrategy website for steps appropriate to your system.

> **Important**
>
> Changes to shared memory configuration can impact all applications and the operating system itself. These steps provide an example; follow them only if you are certain they apply to your environment.

Complete the following steps on each machine:

1. Log in as root, or execute the following command to switch to root:

```
sudo -i bash
```

2. Execute one of the following commands:
   Release 9.0.014 and later:

```
echo "kernel.sem = 250 1024000 250 4096" >> /etc/sysctl.conf
```

   Earlier releases:

```
echo "kernel.sem = 250 32000 32 4096" >> /etc/sysctl.conf
```

3. Execute the following command:

```
echo "vm.max_map_count = 5242880" >> /etc/sysctl.conf
```

4. Reboot the machine.

2. **Set up DNS** — Set up DNS for each machine in your cluster and (using, for example, 'ping', 'host', and 'hostname --fqdn' commands) verify that each machine can resolve itself and each other by DNS name.

3. Ensure that all machines in the Kubernetes cluster have access to the *shared* folder, which is used for internal purposes by MicroStrategy.
   - The shared folder must be accessible to each machine in the Kubernetes cluster. It can use SMB, NFS, a Kubernetes shared volume, or whatever other method of network share your environment permits. For HA deployments, you must create the shared folder on both node and replica hosts, share the folder on the node host, and mount to the previously created folder on the replica host, as read-only.
   - By default, the containers expect the shared folder to be located at **/genesys/gcxi/shared** on the host. Optionally, you can use another location for the shared folder. To do so, complete the following steps prior to container startup: Open the **gcxi.yaml** file for editing, find all instances of the string **genesys/gcxi/shared**, and replace them with the new folder path.

4. Configure RAA — Some Genesys CX Insights reporting features and the associated objects (including certain folders and reports) are not needed in all deployments, or may require additional configuration steps. Beginning with Genesys CX Insights release 9.0.010, the Genesys CX Insights deployment routine automatically enables these reporting features based on the features you enable in RAA. If an RAA feature in the following table is enabled when you deploy Genesys CX Insights (or restart the container), the corresponding feature is enabled in Genesys CX Insights, and relevant folders and reports are visible. To enable or disable one of these features in Genesys CX Insights, you can enable or disable the the corresponding option in RAA, and:
   - In release 9.0.010, restart the Genesys CX Insights container to enable or disable the indicated feature.
   - In release 9.0.011 or later, wait one hour, and Genesys CX Insights automatically commits your changes, enabling or disabling the indicated feature.

   If you disable a feature that has been enabled for a period of time, and there is data in the tables that you wish to retain (for example, to use in reports), copy the tables into the Custom folder before restarting the container (9.0.010) or before an hour has passed (9.0.011).

Configure these RAA options to automatically enable GCXI features

In RAA release 8.5.011.02 and later, a new configuration option, enable-available-features, in the **[agg-feature]** section, enables all features that are supported in the current RAA release, except, in some releases, **enable-gpr-fcr**, as noted below. If you set the **enable-available-features** option, you do not also need to set the individual options here.

If this RAA feature is enabled:	These objects appear in GCXI:
enable-bgs	Chat Bot folder and reports
enable-callback	Callback folder and reports
enable-chat	Chat folder and reports
enable-chat-thread	Chat folder > Chat Thread Report
enable-cobrowse	Co-browse folder and reports
enable-gpr	Predictive Routing folder and reports

enable-gpr-fcr	The Predictive Routing folder, and the following reports: Predictive Routing AB Testing Report, Predictive Routing - AHT & QUEUE. This option is enabled by enable-available-features only in release 8.5.011.02. In later releases, it is not enabled by enable-available-features, and must be enabled manually.
enable-media-neutral	In the Agents folder: Agent Omnichannel Activity Report
enable-sdr	Designer folder and reports
enable-sdr-bot	In the Designer folder: Bot Analytical Dashboard and Final Disposition Dashboard
enable-sdr-survey	In the Designer folder: Survey Answer Report and Survey Statistics Report
eServicesSM	In the Agents folder: Agent Social Engagement Report
post-call-survey	All reports in the Agent folder except details and interval-based reports, and all reports in the Business Results folder.
user-data-gen-dim	All reports in the Agent folder except details and interval-based reports, and all reports in the Business Results, Outbound Contact, and Queue folders.

For information about enabling features in RAA, see the *Reporting and Analytics Aggregates Options Reference* guide.

Contact Center Sizing Categories

Where this document refers to contact centers as *small*, *medium*, or *large*, these terms refer to environments of approximately the sizes listed in the table **Sizing Categories**:

Sizing Categories

Sizing Category	Number of Agents	Number of Agent Groups	Number of Queues	Daily Call Volume
Small	Fewer than 500	Fewer than 50	Fewer than 50	On the order of tens of thousands
Medium	Fewer than 5000	Fewer than 400	Fewer than 1000	Up to 500000
Large	Fewer than 30000	Fewer than 1000	Fewer than 8000	Up to 4000000

Installing Docker and Kubernetes

In most scenarios, you can deploy Docker and Kubernetes by accessing the installation packages and other files directly from the internet (*online scenarios*). However, it is also possible to deploy the software in environments where it is not possible to access the internet or other external networks

from the machines / network where you plan to install Genesys CX Insights (*offline scenarios*). Choose one of the following options:

- Installing Kubernetes and Docker in online scenarios — for most deployments, **Genesys recommends this option**.
- Installing Kubernetes and Docker in offline scenarios — for scenarios where your deployment environment cannot access the internet.

Tomcat Version

GCXI is shipped with the latest Tomcat version available at the moment of the release build.

To find out the Tomcat version included in your version of GCXI, do either of the following:

- Run the following command:
  ```
  docker run --rm <gcxi_image> /opt/tomcat/bin/version.sh
  ```
  For example, `docker run --rm gcxi:100.0.034.0000 /opt/tomcat/bin/version.sh`.

(Or)

- Run the above command through `docker exec` or `kubectl exec` in the running container.
  For example, `kubectl exec <gcxi_running_pod> /opt/tomcat/bin/version.sh`.

> **Tip**
> If you are using a tool other than `docker` or `kubectl`, refer to the vendor's documentation for instructions on how to run the command inside the container.

Installing Kubernetes and Docker in online scenarios

Disclaimer

Genesys is committed to diversity, equality, and inclusivity. This includes using appropriate terms in our software and documentation. Therefore, Genesys is removing non-inclusive terms. For third-party products leveraged by Genesys that include such terms, Genesys uses the following as replacements.

- For the terms master/slave, Genesys uses “primary” and “secondary” or “primary” and “replica,” with exceptions for their use in third-party commands.
- For the terms blacklist/whitelist, Genesys uses blocklist/allowlist.
- For the term master, when used on its own, Genesys uses main wherever possible.

This page describes example steps to prepare a system for the installation of Genesys Customer Experience Insights (Genesys CX Insights), including the installation of Docker and Kubernetes on a Linux server. Use this instructions on this page in environments where it is possible to access the internet or other external networks from the machines / network where you plan to install Genesys CX Insights (*online scenarios*). If your deployment environment cannot access the internet (*offline scenarios*), follow the instructions on Installing Kubernetes and Docker in offline scenarios instead.

For additional information about Docker, see the *Genesys Docker Deployment Guide*.

Important

Notes:

- This page provides an example scenario using Kubernetes release 1.26.3, Docker version 20.10-ce, and cri-dockerd adapter version 0.3.1 to integrate Docker Engine with Kubernetes, with CentOS Linux release 7.9.2009 (Core). Please make sure that your infrastructure components, such as Linux, Kubernetes, CRI engine and Docker (if you use Docker), are of supported versions and have an adequate update cadence.
- Components such as Kubernetes, Docker, and containerd are community-driven products with a very fast upgrade cycle. Installation instructions for these components, provided by this guide, may sometimes not be the most current. We recommend that you double-check the CRI/Docker/Kubernetes installation steps on their corresponding sites.
- This page does not describe all deployment scenarios, and is applicable only to the indicated software release (Operating System, Container Runtime, Kubernetes). For other releases or CRI, the required steps may vary.

Before you begin

Ensure that you have a suitably-prepared environment, as described in Before you install Genesys CX Insights, which must include suitably-prepared hosts (real machines, virtual machines (VM), or cloud instances) with Red Hat Enterprise Linux 7.9 (or a later 7.x release) / CentOS Linux 7.9 (or a later 7.x release) hosts with system suites installed.

Install Docker and Kubernetes

This section describes a typical production deployment of Docker and Kubernetes (K8S), which is an open-source system for automating deployment, scaling, and management of containerized Docker applications, sometimes called a container orchestration system. Before you deploy MicroStrategy and Genesys CX Insights, you must deploy both Docker and Kubernetes. Docker containers and Kubernetes descriptors simplify Genesys CX Insights deployment, and provide flexibility, scalability, and reliability, and simplified future maintenance of Genesys CX Insights.

To deploy Docker and Kubernetes, and configure Kubernetes clusters, follow the Kubernetes installation instructions. The following section provides an example of a Kubernetes deployment process on Red Hat Enterprise Linux 7.9 (or a later 7.x release) / CentOS Linux 7.9 (or a later 7.x release).

Example steps to install Kubernetes

The exact installation procedure for Kubernetes varies significantly depending on numerous factors, including the type of machines in your environment (these could be real machines, virtual machines, or cloud machines), Operating System, networking model, planned load, and Kubernetes version. For information about the exact steps you must follow to install Kubernetes and Docker in your environment, see Kubernetes installation instructions.

The following procedure outlines the steps to follow in one common deployment scenario.

Procedure: Example: Deploying Kubernetes clusters and loading Docker

Purpose: This example procedure illustrates one scenario for the installation of Kubernetes and Docker. For more information:

- Detailed information about the Operating System requirements, see documentation on the Kubernetes web site.

- These instructions are based on Kubernetes documentation.

Prerequisites

- This sample installation is intended for an Red Hat Enterprise Linux 7.9 (or a later 7.x release) / CentOS Linux 7.9 (or a later 7.x release) environment, as described in Before you install Genesys CX Insights. The examples on this page use Docker CE, but Kubernetes supports multiple container runtimes, and you can choose any other Kubernetes-supported engine. Ensure that you have properly installed and configured a supported engine before you proceed.

> **Important**
>
> Some steps can take some time for processing to complete. Genesys does not recommend interrupting any of the processes initiated in this procedure.

Steps

Perform steps 1-9 on each machine, and then subsequent steps as directed.

1. **Prepare hosts** — Prepare CentOS Linux 7.9 (or a later 7.x release) hosts with system suites installed. Earlier 7.x releases of CentOS Linux may work, but have not been tested by Genesys, and may require additional updates or package installation; see the Kubernetes documentation for more information. These can be real machines, virtual machines (VM), or cloud instances.

   > **Important**
   >
   > Changes to shared memory configuration can impact all applications and the operating system itself. These steps

> provide an example; follow them only if you are certain they apply to your environment.

Note the reviewers: Moved "troubleshooting" step to the end of the procedure.

2. Log in with root access.
3. **Install Docker on each machine** by following the instructions in the Docker installation documentation.
4. Execute the following command to verify the Docker CE installation:

   ```
   docker --version
   ```

   The Docker version appears, such as `Docker version 20.10.22.`
5. Complete the following steps to disable swap.
   1. Execute the following command to disable swap for the current session:

      ```
      swapoff -a
      ```

   2. To permanently disable swap, remove the swap partition using `fdisk` or `parted`. Be sure to remove only the swap partition, as removing another partition could cause serious problems. The changes take effect after system restart.
6. Ensure that SELinux is in permissive mode. For example, execute the following commands:

   ```
   setenforce 0
   sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
   ```

7. Ensure that relevant **sysctl** config option are set to 1. For example, execute the following command:

   ```
   cat <<EOF >  /etc/sysctl.d/k8s.conf
   net.bridge.bridge-nf-call-ip6tables = 1
   net.bridge.bridge-nf-call-iptables = 1
   ```

```
net.ipv4.ip_forward=1
EOF
sysctl --system
```

8. **Install kubeadm, kubelet and kubectl** — On each machine, log in as a user having root privileges, (`sudo -i bash`), and execute the following commands:

```
cat <<EOF > /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=0
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF

yum install -y kubelet kubeadm kubectl
```

9. Install cri-dockerd adapter by following the instructions here or follow the steps provided below; On each machine, log in as a user having root privileges, (`sudo -i bash`), and execute the following commands:

```
# install and configure cri-dockerd service
wget https://github.com/Mirantis/cri-dockerd/releases/download/v0.3.1/cri-dockerd-0.3.1-3.el7.x86_64.rpm
yum install cri-dockerd-0.3.1-3.el7.x86_64.rpm

wget https://raw.githubusercontent.com/Mirantis/cri-dockerd/master/packaging/systemd/cri-docker.service
wget https://raw.githubusercontent.com/Mirantis/cri-dockerd/master/packaging/systemd/cri-docker.socket
mv cri-docker.socket cri-docker.service /etc/systemd/system/

systemctl daemon-reload
systemctl enable cri-docker.service
systemctl enable --now cri-docker.socket

# check cri-dockerd socket
systemctl status cri-docker.socket
```

```
# configure the kubelet to use cri-dockerd
a. Open /var/lib/kubelet/kubeadm-flags.env on each affected node (create the file if it does not exist).
b. Modify the --container-runtime-endpoint flag to unix:///var/run/cri-dockerd.sock
i.e. KUBELET_KUBEADM_ARGS="--container-runtime-endpoint=unix:///var/run/cri-dockerd.sock --pod-infra-container-
image=registry.k8s.io/pause:3.9"

# restart the kubelet
systemctl restart kubelet
```

10. Optionally, configure kubectl autocompletion:

    ```
    echo "source <(kubectl completion bash)" >> ~/.bashrc
    ```

Complete the preceding steps on each machine before continuing.

11. **On the Control plane machine only, create a cluster, and deploy the Flannel network:**
    1. Execute the following command to set up a Kubernetes cluster:

       ```
       kubeadm init --pod-network-cidr=10.244.0.0/16 --cri-socket=unix:///var/run/cri-dockerd.sock
       ```

       Note that this command produces large volumes of output, and should include a string similar to: `kubeadm join --token <token> <cpnode-ip>:<cpnode-port> --discovery-token-ca-cert-hash sha256:<hash>` For example: `kubeadm join 10.51.29.20:6443 --token dmijep.e1qmgc4o3sh22pwd --discovery-token-ca-cert-hash sha256:ef846cf825d6234aa7b123723bc312a7ff72a14facf9e3a02bc34a708fb3c877`
       IMPORTANT: This string is required in a later step. Find the string in the output, then copy and save it. Alternatively, redirect command output to a file before completing this step.
    2. Execute the following command to verify the node is running:

       ```
       kubectl get nodes
       ```

       The Control plane node should have a status of NotReady, similar to the following output:

       ```
       gcxi-doc-kube0   NotReady   master    3m        v1.26.3
       ```
    3. Execute the following commands to configure kubectl to manage your cluster:

```
grep -q "KUBECONFIG" ~/.bashrc || {
    echo 'export KUBECONFIG=/etc/kubernetes/admin.conf' >> ~/.bashrc
    . ~/.bashrc
}
```

4. Deploy the Flannel overlay network on the Control plane node machine:
    1. Execute the following command to initiate the Flannel network:

       ```
       kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
       ```

       This may take several minutes to complete; avoid interrupting the process.
    2. Execute the following command to ensure that **kube-dns*** pods (or **coredns**, depending on the release of kubernetes you are using) are running (not **pending** or any other status):

       ```
       kubectl get pods --all-namespaces
       ```

12. **On the worker machines only, join the worker machines to the cluster**.
    Executing the following command (replacing <token>, <primary port>, and <hash> with appropriate values):

    ```
    kubeadm join --token <token> <primary-ip>:<primary-port> --discovery-token-ca-cert-hash sha256:<hash> --cri-socket=unix:///var/
    run/cri-dockerd.sock
    ```

    (or paste in the string you saved in a previous step).

    For example: `kubeadm join 10.51.29.20:6443 --token dmijep.e1qmgc4o3sh22pwd --discovery-token-ca-cert-hash sha256:ef846cf825d6234aa7b123723bc312a7ff72a14facf9e3a02bc34a708fb3c877 --cri-socket=unix:///var/run/cri-dockerd.sock`
13. **On the Control plane node machine, verify nodes:**
    1. Ensure that all nodes are in the **ready** state.
    2. Execute the following command on the Control plane node machine:

       ```
       kubectl get nodes
       ```

    The nodes should have a status of Ready, similar to the following output:

```
gcxi-doc-kube0   Ready     master    3m        v1.26.3
gcxi-doc-kube1   Ready     <none>    22s       v1.26.3
```

Troubleshooting tips

Execute the following troubleshooting commands if you encounter configuration issues with the CentOS Linux packages:

1. This step prevents the error `Requires: container-selinux >= 2.9`:

   ```
   sudo yum install -y http://mirror.centos.org/centos/7/extras/x86_64/Packages/container-selinux-2.107-3.el7.noarch.rpm
   ```

   It may take a little time for this step to complete.

2. This step prevents the error `libtool-ltdl-2.4.2-22.el7_3.x8 FAILED`:

   ```
   sudo yum install http://mirror.centos.org/centos/7/os/x86_64/Packages/libtool-ltdl-2.4.2-22.el7_3.x86_64.rpm
   ```

   A message appears, similar to the following:

   ```
   Total size: 66 k   Installed size: 66 k  Is this ok [y/d/N]:
   ```

   Enter y to continue.

3. This step prevents errors similar to `Package: docker-ce-18.03.1.ce-1.el7.centos.x86_64 (docker-ce-stable) Requires: pigz`:

   ```
   sudo yum install http://mirror.centos.org/centos/7/extras/x86_64/Packages/pigz-2.3.3-1.el7.centos.x86_64.rpm
   ```

   A message appears, similar to the following:

   ```
   Total size: 123 k   Installed size: 123k  Is this ok [y/d/N]:
   ```

   Enter y to continue.

Next Steps

After you have installed and configured Docker and Kubernetes, proceed to Installing Genesys CX Insights.

After completing the steps on this page, complete the following:

- Installing Genesys CX Insights
- Installing report editing software
- Post-Installation steps

Installing Kubernetes and Docker in offline scenarios

Disclaimer

Genesys is committed to diversity, equality, and inclusivity. This includes using appropriate terms in our software and documentation. Therefore, Genesys is removing non-inclusive terms. For third-party products leveraged by Genesys that include such terms, Genesys uses the following as replacements.

- For the terms master/slave, Genesys uses “primary” and “secondary” or “primary” and “replica,” with exceptions for their use in third-party commands.
- For the terms blacklist/whitelist, Genesys uses blocklist/allowlist.
- For the term master, when used on its own, Genesys uses main wherever possible.

This page describes example steps to prepare a system for the installation of Genesys Customer Experience Insights (Genesys CX Insights). Use these instructions in environments where it is not possible to access the internet or other external networks from the machines / network where you plan to install Genesys CX Insights (*offline scenarios*). If your deployment environment has access to the internet (*online scenarios*), use the (simplified) instructions on Installing Kubernetes and Docker in online scenarios.

For additional information about Docker, see the *Genesys Docker Deployment Guide*.

Important

Notes:

- This page provides an example scenario, using Kubernetes release 1.21.2 and Docker version 19.03-ce, on CentOS Linux release 7.5.1804 (Core) on Amazon Web Services (AWS). **Always use the latest approved releases of Kubernetes and Docker.**
- This page does not describe all deployment scenarios, and is applicable to only the indicated software release (Operating System, Docker, Kubernetes). For other releases, the required steps may vary.

Before You Begin

Ensure that you have a suitably-prepared environment, as described in Before you install Genesys CX Insights, which must include suitably-prepared hosts (real machines, virtual machines (VM), or cloud instances) with Red Hat Enterprise Linux 7.5 (or a later 7.x release) / CentOS Linux 7.5 (or a later 7.x release) hosts with system suites installed.

In addition, you require:

- **Online machine** — A machine with online access, which must have:
  - Sufficient space to download the YUM packages.
  - The same operating system version as the offline machines where Genesys CX Insights will be installed. Packages downloaded on a different OS version may not work, because the dependencies that determine which files to download vary depending on the operating system version,

components and packages that already exist on the machine. Failure to follow this recommendation may cause installation on the target machine to fail. If you must use mismatched operating systems, see the YUM and Docker documentation.

- **File transfer** — Some method of transferring files from the online machine to the offline network where you will install Genesys CX Insights.

Download and Install Docker

Use the procedures in this section to download and install Docker.

Procedure: Download Docker (online machine)

Purpose: Use the steps in this procedure to download Docker.

Steps

Complete the following steps on the **online** machine:

1. Execute the following command to configure YUM so it can download packages correctly:

   ```
   yum install -y yum-utils
   ```

2. Execute the following commands to install the Docker repository:

   ```
   yum-config-manager --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo
   ```

3. Execute the following commands to download required files:

   ```
   yumdownloader --assumeyes --destdir=<your_rpm_dir>/yum --resolve yum-utils
   ```

```
yumdownloader --assumeyes --destdir=<your_rpm_dir>/dm --resolve device-mapper-persistent-data
yumdownloader --assumeyes --destdir=<your_rpm_dir>/lvm2 --resolve lvm2
yumdownloader --assumeyes --destdir=<your_rpm_dir>/docker-ce --resolve docker-ce
yumdownloader --assumeyes --destdir=<your_rpm_dir>/se --resolve container-selinux
```

where <your_rpm_dir> is a directory on your online machine.

Procedure: Install Docker (offline machine)

Purpose: Use the steps in this procedure to install Docker.

Steps

Complete the following steps on each machine in the cluster on the **offline** machine (where <your_rpm_dir> is a directory on your offline machine where you placed the files):

1. Copy the Docker files, downloaded in the Download Docker (online machine), from the online machine to the offline machine.
2. Execute the following commands to uninstall any old docker software:

```
yum remove docker \
 docker-client \
 docker-client-latest \
 docker-common \
 docker-latest \
 docker-latest-logrotate \
 docker-logrotate \
 docker-selinux \
```

```
docker-engine-selinux \
docker-engine
```

3. Execute the following command to install yum utilities:

```
yum install -y --cacheonly --disablerepo=* <your_rpm_dir>/yum/*.rpm
```

4. Execute the following commands to install Docker file drivers:

```
yum install -y --cacheonly --disablerepo=* <your_rpm_dir>/dm/*.rpm
yum install -y --cacheonly --disablerepo=* <your_rpm_dir>/lvm2/*.rpm
```

5. Execute the following command to install container-selinux:

```
yum install -y --cacheonly --disablerepo=* <your_rpm_dir>/se/*.rpm
```

6. Execute the following command to install Docker:

```
yum install -y --cacheonly --disablerepo=* <your_rpm_dir>/docker-ce/*.rpm
```

7. Execute the following commands to enable and start docker service:

```
systemctl enable docker
systemctl start docker
```

8. Execute the following commands to verify docker:

```
systemctl status docker
docker version
```

Download and Install Kubernetes

Use the procedures in this section to download and install Kubernetes utilities.

Procedure: Download Kubernetes utilities (online machine)

Purpose: Use the steps in this procedure to download Kubernetes utilities.

Steps

Complete the following steps on the **online** machine:

1. Execute the following commands to install the Kubernetes repository:

```
cat <<EOF > /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
EOF
```

2. Execute the following command to download the Kubernetes utilities:

```
yumdownloader --assumeyes --destdir=<your_rpm_dir> --resolve yum-utils kubeadm-1.21.* kubelet-1.21.* kubectl-1.21.* ebtables
```

Procedure: Install Kubernetes utilities (offline machine)

Purpose: Use the steps in this procedure to install Kubernetes utilities.

Steps

Complete the following steps on each machine in the cluster on the **offline** machine:

1. Copy the Kubernetes utilities files, downloaded in the preceding steps, from the online machine to the offline machine.
2. Execute the following commands to install Kubernetes:

   ```
   yum install -y --cacheonly --disablerepo=* <your_rpm_dir>/*.rpm
   ```

   where <your_rpm_dir> is a directory on your offline machine where you placed the files.
3. Execute the following command to run **kubeadm**, which returns a list of required images:

   ```
   kubeadm config images list
   ```

   A list of the required images appears, similar to the following:

   ```
   k8s.gcr.io/kube-apiserver:v1.21.2
   k8s.gcr.io/kube-controller-manager:v1.21.2
   k8s.gcr.io/kube-scheduler:v1.21.2
   k8s.gcr.io/kube-proxy:v1.21.2
   k8s.gcr.io/pause:3.4.1
   k8s.gcr.io/etcd:3.4.13-0
   k8s.gcr.io/coredns/coredns:v1.8.0
   ```

Download and Install Kubernetes Images

Use the procedures in this section to download and install Kubernetes images.

Procedure: Download Kubernetes Images (online machine)

Purpose: Use the steps in this procedure to download the Kubernetes images.

Steps

Complete the following steps on the **online** machine:

1. **Install Docker on each machine** by following the instructions in the Docker installation documentation.
2. For each image that appears in the list that was returned in the preceding step, execute the following commands to pull the image and save it as a TAR archive:

   ```
   docker pull k8s.gcr.io/<image name>
   docker save k8s.gcr.io/<image name> > <image name>.tar
   ```

   where <image name> is the name of the image to pull, for example:

   ```
   docker pull k8s.gcr.io/kube-apiserver:v1.21.2
   docker save k8s.gcr.io/kube-apiserver:v1.21.2 > kube-apiserver_v1.21.2.tar
   ```

Procedure: Load Kubernetes Images (offline machine)

Purpose: Use the steps in this procedure to load the Kubernetes Images.

Steps

Complete the following steps on each machine in the cluster on the **offline** machine:

1. Copy the Kubernetes images, downloaded in Download Kubernetes Images (online machine), from the online machine to the offline machine.
2. For each image, execute the following command to unpack the image:

   ```
   docker load < <image name>.tar
   ```

   where <image name> is the name of the image to unpack, for example:

   ```
   docker load < kube-apiserver_v1.21.2.tar
   ```

Download and Install Kubernetes Network

Use the procedures in this section to download and install Kubernetes networking files.

Procedure: Download networking files (online machine)

Purpose: Use the steps in this procedure to download Kubernetes networking files.

Steps

Complete the following steps on the **online** machine:

1. Execute the following command to download the yaml descriptor:

   ```
   wget https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
   ```

2. Open the **kube-flannel.yml** file, and find the line that indicates the flannel image version. For example, in the following string, the version is **v0.14.0**:

   ```
   image: quay.io/coreos/flannel:v0.14.0-amd64
   ```

3. Execute the following commands to download and save the image:

   ```
   docker pull quay.io/coreos/flannel:v<version>-amd64
   docker save quay.io/coreos/flannel:v<version>-amd64 > flannel_v<version>-amd64.tar
   ```

   where <version> is the flannel image version you found in the previous step. For example:

   ```
   docker pull quay.io/coreos/flannel:v0.14.0-amd64
   docker save quay.io/coreos/flannel:v0.14.0-amd64 > flannel_v0.14.0_v1.tar
   ```

Procedure: Install Kubernetes networking files (offline machine)

Purpose: Use the steps in this procedure to install Kubernetes networking files.

Steps

Complete the following steps on each machine in the cluster on the **offline** machine:

1. Copy the networking files, downloaded in Download networking files (online machine), from the online machine to the offline machine.
2. Execute the following command to unpack the networking image:

   ```
   docker load < flannel_v<version>_v1.tar
   ```

   where <version> is the version of the flannel software, for example:

   ```
   docker load < flannel_v0.14.0_v1.tar
   ```

Download and Install NGINX

Use the procedures in this section to download and install NGINX.

Procedure: Download NGINX images (online machine)

Purpose: Use the steps in this procedure to download NGINX images.

Steps

Complete the following steps on the **online** machine:

1. Execute one of the following commands to download yaml descriptors:

```
wget https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.30.0/deploy/static/namespace.yaml && \
 wget https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.30.0/deploy/static/rbac.yaml
```

2. Execute the following command to download and save the image:

```
docker pull quay.io/kubernetes-ingress-controller/nginx-ingress-controller:0.30.0
 docker save quay.io/kubernetes-ingress-controller/nginx-ingress-controller:0.30.0 > nginx-ingress-controller_0.30.0.tar
```

Procedure: Load NGINX images (offline machine)

Purpose: Use the steps in this procedure to load NGINX images.

Steps

Complete the following steps on each machine in the cluster on the **offline** machine:

1. Copy the NGINX images, downloaded in Download NGINX images (online machine), from the online machine to the offline machine.
2. Execute the following command to unpack the NGINX image:

```
docker load < nginx-ingress-controller_0.30.0.tar
```

Deploy cluster

Procedure: Example: Deploying Kubernetes cluster

Purpose: This example procedure illustrates one scenario to complete the installation and preparation of Kubernetes and Docker in offline scenarios. For more information, see Kubernetes documentation.

Prerequisites

Ensure that you have a suitably-prepared environment, as described in Before you install Genesys CX Insights.

Steps

Perform steps 1-5 on each machine, and then complete subsequent steps as indicated:

1. Log in with root access.
2. Complete the following steps to disable swap:
    1. Execute the following command to disable swap for the current session:

       ```
       swapoff -a
       ```

    2. To permanently disable swap, remove the swap partition using `fdisk` or `parted`. Be sure to remove only the swap partition, as removing another partition could cause serious problems. The changes take effect after system restart.
3. Ensure that SELinux is in permissive mode. For example, execute the following commands:

   ```
   setenforce 0
   sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
   ```

4. Ensure that the config option **sysctl > net.bridge.bridge-nf-call-iptables** is set to 1. For example, execute the following command:

```
cat <<EOF >  /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
sysctl --system
```

5. Optionally, configure kubectl autocompletion:

```
echo "source <(kubectl completion bash)" >> ~/.bashrc
```

Complete the preceding steps on each PRIMARY and SECONDARY machine before continuing.

6. **On the PRIMARY machine only, create a cluster, deploy the Flannel network, and schedule pods:**
   1. Execute the following command to retrieve the version number for Kubernetes:

      ```
      kubectl version
      ```

      The Kubernetes version is displayed.

   2. Execute the following command to set up a Kubernetes cluster:

      ```
      kubeadm init --pod-network-cidr=10.244.0.0/16 --kubernetes-version=v<version>
      ```

      Where <version> is the version of Kubernetes retrieved in the preceding step. For example:

      ```
      kubeadm init --pod-network-cidr=10.244.0.0/16 --kubernetes-version=v1.21.2
      ```

      Note that this command produces large volumes of output, and should include a string similar to: `kubeadm join --token <token> <primary-ip>:<primary-port> --discovery-token-ca-cert-hash sha256:<hash>` For example: `kubeadm join 10.51.29.20:6443 --token dmijep.e1qmgc4o3sh22pwd --discovery-token-ca-cert-hash sha256:ef846cf825d6234aa7b123723bc312a7ff72a14facf9e3a02bc34a708fb3c877`
      IMPORTANT: This string is required in a later step. Find the string in the output, then copy and save it. Alternatively, redirect command output to a file before completing this step.

   3. Execute the following command to verify the node is running:

```
kubectl get nodes
```

The Control plane node should have a status of NotReady, similar to the following output:

```
gcxi-doc-kube0   NotReady   master    3m        v1.21.2
```

4. Execute the following commands to configure kubectl to manage your cluster:

```
grep -q "KUBECONFIG" ~/.bashrc || {
    echo 'export KUBECONFIG=/etc/kubernetes/admin.conf' >> ~/.bashrc
    . ~/.bashrc
}
```

5. Deploy the Flannel overlay network on the PRIMARY machine:
   1. Execute the following command to initiate the Flannel network:

      ```
      kubectl apply -f <destination path>/kube-flannel.yml
      ```

      Where <destination path>

      This may take several minutes to complete; avoid interrupting the process.

   2. Execute the following command to ensure that **kube-dns*** pods (or **coredns**, depending on the release of kubernetes you are using) are running (not **pending** or any other status):

      ```
      kubectl get pods --all-namespaces
      ```

6. Schedule pods — Note that, for security reasons, the cluster does not schedule pods on the PRIMARY by default. Optionally, to configure the cluster to schedule on the PRIMARY machine, execute the following command on the PRIMARY machine. Executing this command removes the `node-role.kubernetes.io/master` taint from any nodes that have it, so that the scheduler can schedule pods everywhere:

```
kubectl taint nodes --all node-role.kubernetes.io/master-
```

7. **On the SECONDARY machines only, join the SECONDARY to the cluster:**
   Execute the following command (replacing <token>, <primary port>, and <hash> with appropriate values):

```
kubeadm join --token <token> <primary-ip>:<primary-port> --discovery-token-ca-cert-hash sha256:<hash>
```

(or paste in the string you saved in a previous step).

For example: `kubeadm join 10.51.29.20:6443 --token dmijep.e1qmgc4o3sh22pwd --discovery-token-ca-cert-hash sha256:ef846cf825d6234aa7b123723bc312a7ff72a14facf9e3a02bc34a708fb3c877`

8. **On the PRIMARY machine, verify nodes:**
   1. Ensure that all nodes are in the **ready** state.
   2. Execute the following command on the PRIMARY machine:

      ```
      kubectl get nodes
      ```

      The nodes should have a status of Ready, similar to the following output:

      ```
      gcxi-doc-kube0   Ready     master    3m        v1.21.2
      gcxi-doc-kube1   Ready     <none>    22s       v1.21.2
      ```

Next Steps

After you have installed and configured Docker and Kubernetes, proceed to Installing Genesys CX Insights.

Download and Install PostgreSQL Image

Use the procedures in this section to download and install the PostgreSQL image.

Procedure: Download PostgreSQL Image (online machine)

Purpose: Use the steps in this procedure to download the PostgreSQL image.

Steps

Complete the following steps on the **online** machine:

1. Execute the following command pull the image and save it as a TAR archive:

   ```
   docker pull postgres:<version>
   docker save postgres:<version> > postgres.tar.gz
   ```

   where <version> is the PostgreSQL release of the image to pull, for example:

   ```
   docker pull postgres:12
   docker save postgres:12 | gzip > postgres.tar.gz
   ```

Procedure: Load PostgreSQL Image (offline machine)

Purpose: Use the steps in this procedure to load the PostgreSQL Image.

Steps

Complete the following steps on the **offline** machine:

1. Copy the PostgreSQL images, downloaded in Download PostgreSQL Images (online machine), from the online machine to the offline machine.
2. Execute the following command to unpack the image:

```
docker load < postgres.tar.gz
```

Installing Genesys CX Insights - Kubernetes using descriptors

> **Important**
>
> In keeping with Genesys' commitment to diversity, equality, and inclusivity, beginning with release 9.0.019.01, some pod names are changed; this document refers to "gcxi-primary" and "gcxi-secondary" pods. In release 9.0.019.00 and earlier, these pods were named "gcxi-master" and "gcxi-slave".

This page describes the steps required to deploy Genesys Customer Experience Insights (Genesys CX Insights) with Kubernetes, using descriptors, which is the supported and recommended scenario for deploying Genesys CX Insights in most production environments in release 9.0.015 and earlier.

Beginning with release 9.0.016.02, Genesys supports deployment of Kubernetes using Helm, which is recommended for new Kubernetes deployments. This deployment method is suitable for production environments; for other deployment options, see Choose a deployment type and Prerequisites.

Before You Begin

Before you complete the steps on this page, prepare the environment as described in Before you install Genesys CX Insights. In addition:

1. Acquire the Genesys CX Insights installation package

Ensure that you have the latest Genesys CX Insights 9.0 installation packages (IP); talk to your Genesys representative for information about where to download the installation packages.

Installation packages for GCXI

Component	IP / File	tar files*
CustExInsights — Genesys Customer Experience Insights	Docker container (Docker Linux platform) **IP_CustExpInsights_9000XXX_ENU_dockerlinux.zip** (where XXX is the latest release number.)	**gcxi.tar.gz** — contains the **gcxi** Docker image, which contains a fully installed Microstrategy Server 10.x (the latest supported release of MicroStrategy: 11.2.1, for example.). This container provides a *stateless* deployment, where project data (reports, users, and other objects) is stored separately in a MicroStrategy *meta* database. This image is used for production deployments.
	Regular Linux IP (Linux platform)	**data.tar.gz** — contains the various YAML files

	IP_CustExpInsights_9000XXX_ENU_linux.tar.gz (where XXX is the latest release number.)	(Kubernetes script files), such as gcxi.yaml, gcxi-postgres-yaml, and gcxi-init.yaml, and the gcxi.properties file (files which you must edit as part of the deployment procedure), PostgreSQL database dump with MicroStrategy meta-data database for GCXI project, and other files needed for GCXI
CustExInsightOps — Genesys Customer Experience Insights Ops	Docker container (Docker Linux platform) **IP_CustExpInsightsOPS_9000XXX_ENU_dockerlinux.zip** (where XXX is the latest release number.)	**gcxi_control.tar.gz** — contains the **gcxi_control** Docker image, which is used for deployment and configuration of the GCXI solution.
CustExInsightDB — Genesys Customer Experience Insights DB (Discontinued beginning with GCXI release 9.0.019.01)	Docker container (Docker Linux platform) **IP_CustExpInsightsDB_9000XXX_ENU_dockerlinux.zip** (where XXX is the latest release number.)	**gcxi_postgres.tar.gz** — contains the **gcxi_postgres** image, which contains a PostgreSQL database server with GCXI MicroStrategy Project, Meta data, and History databases pre-deployed. This image is discontinued beginning with GCXI release 9.0.019.01
MSSecEntPltf64 — MicroStrategy Secure Enterprise Platform for Windows	MicroStrategy software for Windows (server and client / editing tools) **MicroStrategy_XXX_IntelligentEnterprise_Windows_XXX.zip** (where XX is the current MicroStrategy release. For example, **MicroStrategy_11.3_IntelligentEnterprise_Windows_11.3.0560.0066.zip**.)	MicroStrategy_Secure_Enterprise_Platform_11_3_Win/cpe1705/PI
MSWrkstn — MicroStrategy Workstation	MicroStrategy Workstation software for Windows **workstation-win-ent_XXX.zip** (where XXX is the current MicroStrategy release. For example, **workstation-win-ent_11.3.63.zip**.)	MicroStrategy_Workstation_11.3.63/ The MicroStrategy Workstation package is available beginning with GCXI release 100.0.029.0000

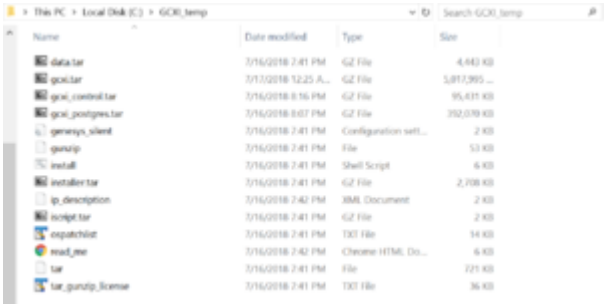
*Important:

In some releases, the names of the container images in the installation package differ from the description in the **Installation packages for GCXI** table. In these scenarios, rename the container images as described in the table **Renaming the images**:

Renaming the images (if your release requires it)

Copy this file from this folder to a convenient location on your local hard drive (for example `C:\GCXI_temp`):	Rename it as:
CustExpInsights ... **dockerlinux**... **9.0.010.04.tar.gz**	**gcxi.tar.gz**
CustExpInsightsDB ... **9.0.010.04.tar.gz** (This image is discontinued beginning with GCXI release 9.0.019.01)	**gcxi_postgres.tar.gz**
CustExpInsightsOps ... **9.0.010.04.tar.gz**	**gcxi_control.tar.gz**

Note that Reporting and Analytics Aggregates (RAA) files are also available in the same location (**Reporting_and_Analytics_Aggregates_G231_850XXXX_ENU_ISO**). See the Reporting and Analytics Aggregates documentation for more information about deploying RAA.



The installation package contents

2. Gather information about data sources

For Genesys CX Insights to produce meaningful reports, you must have installed and properly configured both Genesys Info Mart and Reporting and Analytics Aggregates (RAA):

- Genesys Info Mart release 8.5 database — The Genesys Info Mart documentation describes how to deploy and configure Genesys Info Mart, including information about hardware sizing requirements to support Gensys Info Mart. Genesys CX Insights can provide meaningful reports only if the Info Mart database is regularly populated by a Genesys Info Mart 8.5 application. Genesys Info Mart must be properly configured and installed before Genesys CX Insights runs the aggregation process (RAA). Refer to the *Genesys Info Mart Deployment Guide* or the *Genesys Migration Guide* for information that pertains to configuring, installing, or upgrading Genesys Info Mart.

- Reporting and Analytics Aggregates (RAA) — The RAA documentation describes how to deploy RAA, and how to configure the aggregation process. You must have available all relevant Genesys Info Mart information, including the RDBMS type (Microsoft SQL Server, PostgreSQL, Oracle), hostname, and user credentials.

3. Decide how to handle the meta database

Choose whether to deploy an external PostgreSQL server for the *meta* database:

- Deploying with an external meta database — Requires that you create an external PostgreSQL server on which the MicroStrategy meta database resides.
- Deploying the pre-packaged meta database — Uses a pre-packaged meta database, so that you do not need to deploy or manage a separate PostgreSQL server. This option uses the standard PostgreSQL container. If you choose this option, the PostgreSQL container must have access to its data volume.

Deploying the containers

Use the steps in this section to deploy the Docker containers. Note that Genesys does not ship Docker or Kubernetes as a part of Genesys CX Insights. You must install Docker in your environment before you can load the Genesys CX Insights containers; see Before you install Genesys CX Insights. Install Docker according to the instructions on the Docker site. A Docker deployment provides a complete self-contained environment, so that you do not need to manually configure ports or address compatibility issues.

Procedure: Load Docker images from tar.gz archives

Purpose: Use the steps in this procedure to prepare the Docker containers.

Steps

Complete the following steps on EACH machine, except where noted otherwise:

1. Copy the docker containers (gcxi_control.tar.gz and gcxi.tar.gz) onto each machine in the cluster.
2. Log in as root, or execute the following command to switch to root:

   ```
   sudo -i bash
   ```

3. On each machine in the cluster, execute the following command:

   ```
   docker load < gcxi_control.tar.gz
   ```

4. On each machine in the cluster, execute the following command:

   ```
   docker load < gcxi.tar.gz
   ```

Procedure: Retag Images

Purpose: Images can contain tagging strings that cause installation errors. Use the steps in this procedure to ensure that images have proper tagging.

Prerequisites

- In release 9.0.011 and later, this procedure applies to all image files, whether downloaded from a repository, or from tar.gz files.

Steps

1. Execute the following command to verify that the images loaded correctly:

   ```
   docker images
   ```

   The console lists the installed Docker images.

   ```
   $ docker images
   REPOSITORY                                                TAG                 IMAGE ID            CREATED             SIZE
   pureengage-docker-production.jfrog.io/gcxi/gcxi_control   9.0.013.01          761b48dfa69a        6 weeks ago         1.32GB
   pureengage-docker-production.jfrog.io/gcxi/gcxi           9.0.013.01          e7a7216f2f2f        6 weeks ago         11.7GB
   pureengage-docker-production.jfrog.io/gcxi/gcxi_postgres  9.0.013.01          068b8c6ba06c        6 weeks ago         3.53GB
   ```

2. Execute the following commands to retag each of the images:

   ```
   docker tag <REPOSITORY>/gcxi:<RELEASE> gcxi:<RELEASE>
   ```

   ```
   docker tag <REPOSITORY>/gcxi_postgres:<RELEASE> gcxi_postgres:<RELEASE>
   ```

   ```
   docker tag <REPOSITORY>/gcxi_control:<RELEASE> gcxi_control:<RELEASE>
   ```

   where:

   <REPOSITORY> is the identifier for the repository from which you downloaded the files.

   <RELEASE> is the a string corresponding to the release you are installing (such as 9.0.014.02),

   For example:

   ```
   docker tag pureengage-docker-production.jfrog.io/gcxi/gcxi:9.0.014.02 gcxi:9.0.014.02
   ```

   ```
   docker tag pureengage-docker-production.jfrog.io/gcxi/gcxi_postgres:9.0.014.02 gcxi_postgres:9.0.014.02
   ```

   ```
   docker tag pureengage-docker-production.jfrog.io/gcxi/gcxi_control:9.0.014.02 gcxi_control:9.0.014.02
   ```

3. Execute the following command to verify that the images loaded correctly, and have correct tagging:

   ```
   docker images
   ```

   The console lists the installed Docker images. Compare the result to the figure **Docker Images**; each image must have a name in the REPOSITORY column *with no preceding path*, and a value in the TAG column that corresponds to the release. Note that each image appears twice in the list, this is expected behavior, because each one has two tags.

Procedure: Install the CustExInsights component

Purpose: Use the steps in this procedure to install the Genesys Customer Experience Insights Linux (CustExInsights) component. This procedure applies to release 9.0.009 and later.

Steps

Complete the following steps on the PRIMARY machine:

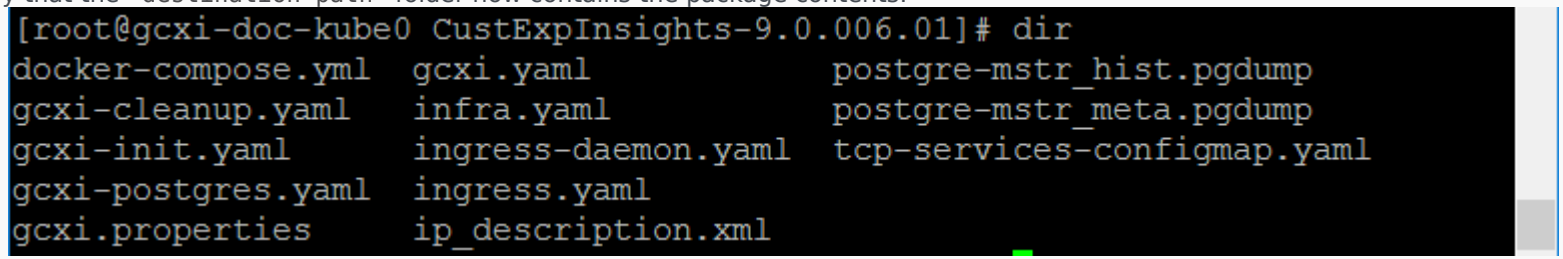
1. Download the CustExpInsights Installation Package (IP), extract its contents, and copy the `\9.0.00x.0x\linux\ip` subfolder to a directory on the PRIMARY machine.
2. Use the following commands to install the software automatically:
   1. Execute the following command to set permissions on the installation script:

      ```
      chmod a+x install.sh
      ```

   2. Execute the installation script:

```
./install.sh
```

3. Verify that the <destination path> folder now contains the package contents:



Tip

To identify the folder where the Genesys CX Insights files are installed, execute the following command:

```
sudo find -name gcxi.properties
```

which returns the path to the `gcxi.properties` file, such as:

```
/genesys/gcxi/gcxi.properties
```

Procedure: Enter database information in the properties file

Purpose: Use the steps in this procedure to populate the **gcxi.properties** file with information about your Info Mart. This procedure is intended for Genesys CX Insights release 9.0.010 and later.

Steps

On the PRIMARY machine, open the **gcxi.properties** file for editing, and edit it as follows:

Genesys CX Insights database properties

This section of the file provides information about defining data sources. Two methods are supported: DSNDEF variables and an externally mounted **obdc.ini** file (supported in GCXI release 9.0.015 and later) — choose one of the two methods. Alternatively, if you don't populate define a data source (using either DSN variables or an OBDC file, as discussed) Genesys CX Insights uses the demo database provided in the container.

Defining data sources using DSNDEF variables

Beginning with release 9.0.010, Genesys CX Insights supports more than one project, and allows you to define multiple DSNs, as follows:

1. Define a **DSNDEF** variable for each DSN. The name of the variable must consist of the string DSNDEF, following by one or more unique characters, for example DSNDEF1, followed by the relevant properties, in the following format:

   ```
   DSNDEF1=DRV_TYPE=<DRV_TYPE>;GCXI_QENGINE=<GCXI_QENGINE>;DSN_NAME=<DSN Name>;
    DB_TYPE=<DB_TYPE>;DB_TYPE_EX=<DB_TYPE_EX>;HOST=<host>;
    PORT=<PORT>;ORCL_SNAME=<ORCL_SNAME>;ORCL_SID=<ORCL_SID>;
    DB_NAME=<DB_NAME>;LOGIN=<USERNAME>;PASSWORD=<PASSWORD>;ENCODING=<UTF_OPTION>
   ```

   where:

   <**DRV_TYPE**> (OPTIONAL) is a value that controls the MSTR setting for the corresponding Database Connections object. Consider setting this

option if you are troubleshooting with connection to database. The following values are supported: `ODBC`, `JDBC`. Default value: `ODBC`. For example `DRV_TYPE=JDBC`.

<**GCXI_QENGINE**> (OPTIONAL) is a value that controls the MSTR setting for the corresponding Database Connections object. This option works only when DRV_TYPE is configured to `JDBC`. Consider setting this option if you are troubleshooting the performance of project reports. The following values are supported: `ON`, `OFF`. Default value: `OFF`. For example `GCXI_QENGINE=ON`.

<**DSN_NAME**> is the DSN type. The following values are supported: GCXI_GIM_DB or IWD_DB

<**DB_TYPE**> is the RDBMS type. The following values are supported: POSTGRESQL, SQLSERVER, ORCLW

<**DB_TYPE_EX**> is your extended RDBMS type. For an up-to-date list of supported values, open the file **$MSTR_INSTALL_HOME/install/DATABASE.PDS**, and check the **DSSOBJECT** element, **NAME** attribute. For example, the following values are supported at the time of writing, but may vary depending on the release you are installing: 'Microsoft SQL Server 2012' 'Microsoft SQL Server 2014' 'Microsoft SQL Server 2016' 'POSTGRESQL' 'Oracle 12cR2' 'Oracle 18c' 'Oracle 19c'

Note, however, that there is a known issue with `Oracle 11g`; Genesys recommends that you use `Oracle 11gR2` instead.

<**PORT**> is the Genesys Info Mart or iWD RDBMS port number.

<**DB_LOGIN**> is a Microstrategy object with $DSN_NAME.

For Oracle deployments, populate either **ORCL_SID** or **ORCL_SNAME** (not both):

- <**ORCL_SNAME**> is the Oracle Service Name, an alias that is used to remotely connect to the database.
- <**ORCL_SID**> is the Oracle SID, a unique name that identifies the instance / database.

<**DB_NAME**> is the actual Genesys Info Mart or iWD database name (populate this only for Microsoft SQL and PostgreSQL deployments).

<**USERNAME**> is the Genesys Info Mart or iWD administrator user name.

<**PASSWORD**> is the Genesys Info Mart or iWD password.

<**UTF_OPTION**> (OPTIONAL) is the value that controls the MSTR setting for the corresponding Database Connections object in release 9.0.013 and earlier; **UTF_OPTION** is not used in release 9.0.014 and later.

OR

Define data sources using ODBC

Beginning with release 9.0.015, Genesys CX Insights supports the use of a custom external **obdc.ini** file to define the database. To use this method instead of DSN variables, complete the following steps:

1. Define the variable **SKIP_DSN**, to disable automatic DSN configuration. You can define this variable globally or for individual DSNDEF.
2. Create an externally mounted **odbc.ini** file similar to the following:

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: gcxi-config-odbc
  namespace: genesys
data:
  DSN_ODBCINI_OVERRIDE: |-
    [ODBC Data Sources]
    GCXI_GIM_DB=MicroStrategy ODBC Driver for PostgreSQL Wire Protocol
    IWD_DB=MicroStrategy ODBC Driver for PostgreSQL Wire Protocol

    [GCXI_GIM_DB]
    ApplicationUsingThreads=8
    AlternateServers=8

    [IWD_DB]
    ApplicationUsingThreads=9
    Description=MicroStrategy ODBC Driver for PostgreSQL Wire Protocol
    FailoverPreconnect=9
```

   This example uses **DSN_ODBCINI_OVERRIDE**. Alternatively, use DSN_ODBCIN instead of DSN_ODBCINI_OVERRIDE to override only the sections [GCXI_GIM_DB] and [IWD_DB] in the **odbc.ini** file (in all pods).
3. Define the following variable in the **configmap yaml** file:

- **DSN_ODBCINI** — define the entire definition for a particular DSN.

MicroStrategy database properties This section of the file provides information about MicroStrategy *meta* and *history* database properties. These are databases where Microstrategy stores internal information, and are created automatically during Genesys CX Insights deployment. You can choose whether to deploy an external PostgreSQL server for the meta database:

- Deploying with an external meta database — Requires that you create an external PostgreSQL server on which the MicroStrategy meta database resides.
- Deploying the pre-packaged meta database — Uses a pre-packaged meta database, so that you do not need to deploy or manage a separate PostgreSQL server. This option uses the standard PostgreSQL container. If you choose this option, the PostgreSQL container must have access to its data volume, so note the following two possible configurations:
  - PostgreSQL data volume is shared across worker nodes. In this case no additional steps needed, and the PostgreSQL container can run on any worker node,
    OR
  - If PostgreSQL data volume resides on a specific node, you must tie the PostgreSQL container execution to this node by completing the following steps:
    1. Choose a worker node for your PostgreSQL container, and apply a label to the node using the command `kubectl label nodes`.
       For example:

       ```
       kubectl label nodes gcxi/role=gcxi-db
       ```

    2. Open the **gcxi-postgres.yaml** file for editing.
    3. Uncomment the `## nodeSelector example ##` section, and edit it as described in the comments, so that **node-selector** is set to **worker node**. Ensure that the label in nodeSelector matches the one you applied to the node.
       For example:

       ```
       gcxi/role: gcxi-db
       ```

       For more information, see Assign Pods to Nodes on the Kubernetes website.

- Postgres containers store database data in a Docker volume, which is a physical directory on the host that is mapped to the container. By default, this directory is `/genesys/gcxi/data`. Prior to starting the container, you can change the directory that is used to store the data by editing the **gcxi-postgres.yaml.** file, and changing the `/genesys/gcxi/data` to another valid path.

> **Important**
>
> In scenarios where you use the pre-packaged meta database, Genesys strongly recommends that you regularly back up the contents of the **Docker volume** directory, as it contains your Genesys CX Insights database data.

Choose one of the following:

- If you plan to use the pre-packaged meta database, leave all the `META*` properties empty.
- If you are using an external PostgreSQL server to host the meta database, complete the following steps:

1. Populate the `META_DB_ADMIN`, `META_DB_ADMINDB`, and `META_DB_ADMINPWD` properties with an existing user name, database name, and password. The user name specified must correspond to an account that has the necessary permissions to create databases and database users, and to assign ownership.
   For example: `META_DB_ADMIN=postgres`, `META_DB_ADMINDB=postgres_db`, and `META_DB_ADMINPWD=postgres_pwd`.
2. Populate the `META_DB_HOST` and `META_DB_PORT` properties host and port where the meta and history databases will be created:

   ```
   META_DB_HOST=<Host name> and META_DB_PORT=<port>
   ```

   where <Host name> is the host name where the external PostgreSQL server is located, and <port number> is the port of the external PostgreSQL server (usually 5432).
3. Populate the `META_DB_LOGIN` and `META_DB_PASSWORD` properties with credentials for a new user account to be created for the MicroStrategy meta database. The deployment routine uses the information you provide to create a new user account for the meta database.
   For example: `META_DB_LOGIN=mstr_meta_kube` and `META_DB_PASSWORD=g1n2s3s4`

4. Populate the `META_HIST_LOGIN` and `META_HIST_PASSWORD` properties with appropriate credentials for the MicroStrategy history database. The deployment routine uses the information you provide to create a user account for the history database.
   For example: `META_HIST_LOGIN=mstr_hist_kube` and `META_HIST_PASSWORD=g1n2s3s4`.

Other properties

This section of the file provides information about other relevant properties:

- Populate the `MSTR_PASSWORD` property with a suitable administrative password (minimum 8 characters, with 1 each of upper case, lower case, and numeric). The deployment routine uses the information you provide to set the administrator password for Microstrategy.
  For example: `MSTR_PASSWORD=Pa55word`. You can change this later, using the steps in Changing administrator passwords.
- Ensure that the **gcxi.properties** > `GCXI_VERSION` property was set correctly by the installer. It must correspond to the release you are installing, for example `GCXI_VERSION=9.0.014.02`.
- Beginning with release 9.0.013, a new variable (`TOMCAT_GCXIROOT=<false|true>`) allows you to optionally redirect the base URL to provide a shorter form of the URL used to access reports, in the form of `http://<server>` instead of `http://<server>:8080/MicroStrategy`.
  To enable redirect, set the container variable to `true`:

  ```
  TOMCAT_GCXIROOT=true
  ```

Procedure: Deploy Genesys CX Insights

Purpose: Use the steps in this procedure to deploy Genesys CX Insights into Kubernetes.

Steps

Complete the following steps on the PRIMARY machine:

1. To create Kubernetes namespace 'genesys', execute the following command:

   ```
   kubectl create -f <destination path>/infra.yaml
   ```

   where <destination path> is the folder in which the Genesys Installation Package (IP) is stored, for example:

   ```
   kubectl create -f /genesys/gcxi/infra.yaml
   ```

2. To set 'genesys' namespace as the default namespace, execute the following command:

   ```
   kubectl config set-context $(kubectl config current-context) --namespace=genesys
   ```

3. To load the variables into Kubernetes, execute the following command:

   ```
   kubectl create configmap gcxi-config --from-env-file=<destination path>/gcxi.properties --namespace genesys
   ```

   where <destination path> is the folder in which the Genesys IP is stored, for example:

   ```
   kubectl create configmap gcxi-config --from-env-file=/genesys/gcxi/gcxi.properties --namespace genesys
   ```

4. If you are using secrets to store data (as discussed in Configure secrets), execute the following command to create the secrets:

   ```
   kubectl create -f <destination path>/gcxi-secrets.yaml
   ```

5. If you are hosting the meta database on an external server, skip this step. If you are using the pre-packaged meta database, complete the following steps:

   1. Execute the following command to start the PostgreSQL database container, which is required so that your GCXI meta database to run in the PostgreSQL container as a part of the Kubernetes cluster:

      ```
      kubectl create -f <destination path>/gcxi-postgres.yaml
      ```

where <destination path> is the folder in which the Genesys IP is stored, for example:

```
kubectl create -f /genesys/gcxi/gcxi-postgres.yaml
```

2. Execute the following command to verify the state of the gcxi-postgres pod:

   ```
   kubectl get pods | grep 'gcxi-postgres*'
   ```

   The pod status should be Running, for example:

   ```
   gcxi-postgres-5cd4d45754-mss6p   1/1       Running     0          6d
   ```

   If it has any other state, wait a few minutes and check again (it may take some time).

6. To create the MicroStrategy meta database, complete the following steps:

   1. Execute the following command to create the database:

      ```
      kubectl create -f <destination path>/gcxi-init.yaml
      ```

      For example:

      ```
      kubectl create -f /genesys/gcxi/gcxi-init.yaml
      ```

      where <destination path> is the folder in which the Genesys IP is stored. Note that this step is required even if you use the pre-packaged PostgreSQL container.

      Troubleshooting Tip: When creating the meta database, if you encounter an error similar to *Could not translate host name "gcxi-postgres" to address*, see the Troubleshooting page.

   2. Execute the following command to verify the state of the gcxi-init pod:

      ```
      kubectl get pods | grep 'gcxi-init*'
      ```

      The pod status should be Completed, for example:

      ```
      gcxi-init-l4b4x                  0/1       Completed   0          6d
      ```

If it has any other state, wait a few minutes and check again (it may take some time).

7. To deploy the MicroStrategy containers, execute the following command:

   ```
   kubectl create -f <destination path>/gcxi.yaml
   ```

   where <destination path> is the folder in which the Genesys IP is stored, for example:

   ```
   kubectl create -f /genesys/gcxi/gcxi.yaml
   ```

8. Complete the following steps to verify that both GCXI pods are running:
   1. Execute the following command to verify the state of the PRIMARY pod:

      ```
      kubectl get pods | grep 'gcxi-primary*'
      ```

      The pod status should be Running, for example:

      ```
      gcxi-primary-549f6897f-zghqf      1/1       Running     0          6d
      ```

      If it has any other state, wait a few minutes and check again (it may take some time).
   2. Execute the following command to verify the state of the secondary gcxi pod:

      ```
      kubectl get pods | grep 'gcxi-secondary*'
      ```

      The pod status should be Running, for example:

      ```
      gcxi-secondary-75fdb444df-z5nbq      1/1       Running     0          6d
      ```

      If it has any other state, wait a few minutes and check again (it may take some time).

> **Important**
> The MicroStrategy server instance that runs in the container includes a temporary pre-activated key, which is required for the operation of MicroStrategy. Request a replacement key from your Genesys account representative; you need the new key to complete the procedure Install a new license key.

Configuring Ingress

By default, Kubernetes does not expose any app ports publicly. To make your app accessible, you must configure a special entity called *Ingress*. As for any Kubernetes entity, a variety of Ingress methods are supported, for more information see Kubernetes documentation. The following sections provide examples of a simple case where an NGINX daemon is run on each cluster node.

Because of differences in supported Kubernetes versions, the instructions vary depending on the release of Genesys CX Insights you have deployed (see the Product Alert for information about release support); be sure to use the instructions that are suitable for your deployment.

Procedure: Configuring Ingress in release 100.0.024 and later

Purpose: Use the steps in this example procedure to configure Ingress on selected ports, for Genesys CX Insights release 100.0.024 and later. This release requires different installation steps than used in previous releases. This procedure includes steps to remove the old controller if it was installed in a previous release.

If you are upgrading from an earlier release where you installed Ingress, you can either:

- Skip this procedure, and continue to use the old Ingress controller.
  OR

- Complete this procedure, which includes steps to remove the old controller before installing the new one.

Prerequisites

Ensure that the latest Helm version is installed on the PRIMARY machine. This procedure applies to Helm 3.7.1 or later, and NGINX Ingress controller 1.0.4 or later.

Steps

Complete the following steps on the PRIMARY machine:

1. If you previously deployed Ingress under Genesys CX Insights 9.0.012 through 100.0.023, delete the old ingress:

   ```
   kubectl delete -f <ip_path>/nginx-configmap.yaml -n ingress-nginx
   kubectl delete -f <ip_path>/nginx-daemon.yaml -n ingress-nginx
   ```

   where <ip_path> is the folder in which the previously-installed Genesys IP is stored.

2. If an error similar to `Internal error occurred: failed calling webhook...` sometimes appears, you must execute the following command to delete the legacy webhook:

   ```
   kubectl delete ValidatingWebhookConfiguration gcxi-nginx-ingress-nginx-admission
   kubectl delete namespace ingress-nginx
   ```

3. To deploy nginx-ingress controller, execute the following commands:

   ```
   helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
   helm repo add stable https://charts.helm.sh/stable
   helm repo update

   helm install ingress-nginx ingress-nginx/ingress-nginx --set
   controller.hostNetwork=true,controller.hostPort.enabled=true,controller.kind=DaemonSet,tcp.34952=genesys/
   gcxi:mstr,tcp.8180=genesys/gcxi:metrics -n ingress-nginx --create-namespace
   ```

4. Create gcxi ingress rules:

   ```
   kubectl apply -f <destination path>/gcxi-ingress.yaml
   ```

Procedure: Configuring Ingress in release 100.0.023 and earlier

Purpose: Use the steps in this example procedure to configure Ingress on selected ports. This release supports Ingress controller 0.30.0, which requires different installation steps than used in previous releases. This procedure is valid for release 9.0.012.01 through 100.0.023.

If you previously deployed Ingress in an earlier release, you can either:

- Skip this procedure, and continue to use the old Ingress controller.
  OR
- Complete this procedure, which includes steps to remove the old controller before installing the new one.

Steps

Complete the following steps on the PRIMARY machine:

1. If you previously deployed Ingress under Genesys CX Insights 9.0.011.02 or earlier, delete the old ingress:

   ```
   kubectl delete -f <destination path>/ingress.yaml
    kubectl delete -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.20.0/deploy/namespace.yaml
   ```

   where <destination_path> is the folder in which the Genesys IP (for release 9.0.011.02 or earlier) is stored.

2. Deploy ingress controller infrastructure — complete one of the following steps:
   - In **online** deployment scenarios, execute the following commands:

     ```
     kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.30.0/deploy/static/namespace.yaml
     kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.30.0/deploy/static/rbac.yaml
     ```

     OR

   - In **offline** deployment scenarios, complete the following steps:
     1. Copy the **namespace.yaml** and **rbac.yaml** files from the online machine to the offline machine, placing them in a convenient folder, such as the the folder in which the Genesys IP is stored.
     2. Execute the following commands on the offline machine:

        ```
        kubectl apply -f <path>/namespace.yaml
        kubectl apply -f <path>/rbac.yaml
        ```

        where <path> is the folder on the offline machine in which you stored the **namespace.yaml** and **rbac.yaml** files.
3. Deploy ingress controller configmaps — execute the following command to deploy ingress tcp rules:

   ```
   kubectl apply -f <destination path>/nginx-configmap.yaml
   ```

   where <destination path> is the folder in which the Genesys IP is stored.

   The **nginx-configmap.yaml** file contains the port value for non-HTTP traffic. This port is 34952 and it is used by MicroStrategy client tools, such as MicroStrategy Developer.

   For example:

   ```
   kubectl apply -f <destination path>/nginx-configmap.yaml
   ```
4. Define hostPort values:

   ```
   kubectl apply -f <destination path>/nginx-daemon.yaml
   ```

   The hostPort values specified in the **nginx-daemon.yaml** file are the ports to which your http, https, and mstr-tcp traffic will be exposed. The mstr-tcp port is used internally by

Microstrategy applications, such as Developer.

5. Create gcxi ingress rules:

```
kubectl apply -f <destination path>/gcxi-ingress.yaml
```

Configure secrets

Procedure: Configuring Kubernetes Secrets

Purpose: Use the steps in this procedure to optionally configure Kubernetes Secrets. When this feature is configured, Kubernetes stores configuration data, such as passwords, in secure Kubernetes objects. If this feature is not configured, such information is stored in plain text.

Steps

1. Open the **gcxi.properties** and **gcxi-secrets.yaml**, and review instructions and examples in the comments.
2. Configure the following variables for the secrets that are required in your environment:

```
GCXI_GIM_DB.DB_NAME: <base64 value>
 GCXI_GIM_DB.LOGIN: <base64 value>
 GCXI_GIM_DB.PASSWORD: <base64 value>
 IWD_DB.DB_NAME: <base64 value>
```

```
  IWD_DB.LOGIN: <base64 value>
  IWD_DB.PASSWORD: <base64 value>
  MSTR_PASSWORD: <base64 value>
  META_DB_ADMIN: <base64 value>
  META_DB_ADMINDB: <base64 value>
  META_DB_ADMINPWD: <base64 value>
  META_DB_PASSWORD: <base64 value>
  META_HIST_PASSWORD: <base64 value>
  TOMCAT_ADMINPWD: <base64 value>
```

Where <base64 value> is the base64-encoded values for the variable.

Note that "GCXI_GIM_DB" and "IWD_DB" have corresponding DSNDEF entries in the **gcxi.properties** file.

Ensure that the DSNDEF variables used in the secrets correspond to entries in the **gcxi.properties** file, for example:

- Secrets:

```
GCXI_GIM_DB.DB_NAME: bXlfZGI=
GCXI_GIM_DB.LOGIN: bXlfbG9naW4=
GCXI_GIM_DB.PASSWORD: bXlfcGFzc3dvcmQ=
```

- Corresponding DSNDEF entry in **gcxi.properties** (Note the presence of DB_NAME, LOGIN, PASSWORD):

```
DSNDEF1=DSN_NAME=GCXI_GIM_DB;DB_TYPE=POSTGRESQL;DB_TYPE_EX=PostgreSQL;HOST=gi2-qadb;PORT=5432;DB_NAME=;LOGIN=;PASSWORD=;
```

Note that, if any variable is present in both **gcxi-secrets.yaml** and **gcxi.properties**, the value from **gcxi-secrets.yaml** is used.

Verify the installation

Once all steps are complete, use the information in this section to verify that the installation was successful.

The installed Genesys CX Insights folder

Procedure: Verifying Genesys CX Insights installation

Purpose: Use the steps in this procedure to verify the installation. The Genesys CX Insights installation routine creates the folder shown in the figure *The installed Genesys CX Insights folder*.

Steps

After you have successfully installed Genesys CX Insights, complete the following steps:

1. Execute the following command and examine the output:

   ```
   kubectl get nodes
   ```

   The output should be similar to the following:

   ```
   NAME             STATUS    ROLES     AGE       VERSION
    spb-rhel-mstr1   Ready     primary   6d        v1.18.1
    spb-rhel-mstr2   Ready     <none>    6d        v1.18.1
   ```

2. Execute the following command and examine the output:

   ```
   kubectl get pods
   ```

   The output should be similar to the following:

   ```
   NAME                            READY     STATUS      RESTARTS   AGE
    gcxi-init-2qvcd                 0/1       Completed   0          6d
    gcxi-primary-587dc679c-fn2w4     1/1      Running     0          6d
    gcxi-postgres-77b7f946c-drck4   1/1       Running     0          6d (this line appears only if prepackaged PostgreSQL server
   is used)
    gcxi-secondary-5d9f4485bb-d8v25     1/1   Running     1          6d
   ```

3. Execute the following command and examine the output:

   ```
   kubectl get services
   ```

   The output should be similar to the following:

   ```
   NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)              AGE
    gcxi            ClusterIP   10.98.156.54    <none>        34952/TCP,8080/TCP   6d
    gcxi-postgres   NodePort    10.101.64.127   <none>        5432:31642/TCP       6d (this line appears only if prepackaged
   PostgreSQL server is used)
   ```

```
mstr-01         ClusterIP   None            <none>        34952/TCP,8080/TCP   6d
mstr-02         ClusterIP   None            <none>        34952/TCP,8080/TCP   6d
```

4. View the Genesys CX Insights reports in MicroStrategy Web to confirm that the reports are installed, by pointing your web browser to `http://<servername>:<port>/MicroStrategy/servlet/mstrWeb`, where `<servername>` is the IP or host name of any worker node, and `<port>` is the port number (usually `80`).
5. Verify the schema version.
6. Verify the Genesys CX Insights Release number.
7. View the GCXI Project in MicroStrategy Developer.

> **Important**
>
> Unlike most other Genesys applications, Genesys CX Insights is not configured as an application within Genesys Configuration Server, nor is it started (or stopped) by using the Genesys Solution Control Interface.

Procedure: Install a new License key

Purpose: Use the steps in this procedure to install a new license key. The MicroStrategy server instance that runs in the container includes a temporary pre-activated key, which is required for the operation of MicroStrategy.

Prerequisites

Obtain a new license key; contact your Genesys Customer Care representative for assistance.

Steps

1. Execute the following command to back up the GCXI meta db:

   ```
   kubectl apply -f <destination path>/gcxi-backup.yaml
   ```

   where <destination path> is the folder in which the Genesys IP is stored, for example:

   ```
   kubectl apply -f /genesys/gcxi/gcxi-backup.yaml
   ```

2. Execute the following commands to stop currently running containers:

   ```
   kubectl scale deploy/gcxi-secondary --replicas=0
   kubectl scale deploy/gcxi-primary --replicas=0
   ```

3. Edit the **gcxi.properties** file, and add the line

   ```
   MSTR_LICENSE=<your new license>
   ```

   where <your new license> is the new license key value

   This adds the MSTR_LICENSE environment variable to your Genesys CX Insights environment.

4. Execute the following commands to load gcxi.properties into Kubernetes:

   ```
   kubectl delete configmap gcxi-config
   kubectl create configmap gcxi-config --from-env-file=<destination path>/gcxi.properties --namespace genesys
   ```

   where <destination path> is the folder in which the Genesys IP is stored, for example:

```
kubectl create configmap gcxi-config --from-env-file=/genesys/gcxi/gcxi.properties --namespace genesys
```

5. Execute the following command to start the PRIMARY container:

   ```
   kubectl scale deploy/gcxi-primary --replicas=1
   ```

   Wait until PRIMARY is done (wait until Tomcat is up, and MicroStrategyWeb page is available).

6. Execute the following command to start the SECONDARY container:

   ```
   kubectl scale deploy/gcxi-secondary --replicas=1
   ```

7. Optionally, verify that the new License key is installed by checking container's log file (pod stdout). For more information, see Generating logs.

Keep in mind that you must perform additional post-installation setup steps before actively using the reports and projects. After completing the steps on this page, complete the following:

- Installing report editing software
- Post-Installation steps

Installing Genesys CX Insights - Kubernetes using Helm

Beginning with release 9.0.016, Genesys CX Insights supports deployment using Helm Charts on Kubernetes clusters. Helm Charts provide an alternative to Kubernetes descriptors. This deployment method is suitable for production environments; for other deployment options, see Choose a deployment type and Prerequisites.

This is an example scenario — This page provides a high-level outline, illustrating one scenario to help you visualize the overall process. Genesys does not provide support for Helm or other third-party products, so you must have knowledge of Helm and other products to complete this type of installation. This example is suitable for a small deployment using Kubernetes clusters on RedHat Enterprise Linux; the steps for CentOS are similar. GCXI is known to work with Helm-3, which is described on this page.

Prerequisites for deploying using Helm

Before you begin, ensure that:

- Your Kubernetes cluster is configured and running in a suitable environment, with nodes in the **Ready** state, as described in Kubernetes documentation.
- Helm-3 is installed on the Control plane node, as described in the Helm documentation.
- The images **gcxi** and **gcxi_control** are loaded and tagged on each worker node.
- On each worker node, values are set for `kernel.sem`, `vm.max_map_count`, as required by MicroStrategy. For example:

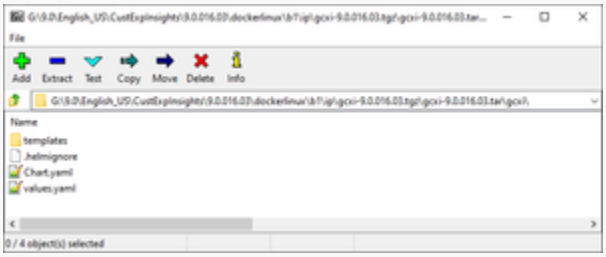
```
echo "kernel.sem = 250 1024000 250 4096" >> /etc/sysctl.conf
echo "vm.max_map_count = 5242880" >> /etc/sysctl.conf
sysctl -p
```

Deploying GCXI with Helm-3

The following procedures describe example steps to deploy GCXI with Helm. The exact steps required will vary for your environment.

Procedure: 1. Preparing for installation

Purpose: Prepare the environment, and gather files needed for deployment.



Contents of the Helm IP

Prerequisites

Within the the Genesys Customer Experience Insights package for the Docker Linux OS, look for the Helm installation package (IP) — a small TGZ file (for example **gcxi-9.0.018.00.tgz**) that contains the Helm files. You require these files to complete this procedure.

Steps

1. On the Control plane node, create a folder: **helm**.
2. Copy the Helm installation package (for example **gcxi-9.0.018.00.tgz**) into the **helm** folder, and extract the archive into a subfolder called **helm/gcxi**.
3. View the file **helm/gcxi/Chart.yaml**, and ensure that the appVersion is set to the desired GCXI version.
4. Open the file **helm/gcxi/values.yaml**, and follow the instructions it provides to guide you in creating a new file, **values-test.yaml** with appropriate settings. Save the new file in the **helm** folder.
   For example, the following content in the **values-test.yaml** file is appropriate for a simple deployment using PostgreSQL inside the container, and PersistentVolumes of the type **local**. Create appropriate content in the **values-test.yaml** file for your environment:

```
gcxi:
  env:
    GCXI_GIM_DB:
      DSNDEF:
DSN_NAME=GCXI_GIM_DB;DB_TYPE=POSTGRESQL;DB_TYPE_EX=PostgreSQL;HOST=gim_db_host;PORT=5432;DB_NAME=gi
      LOGIN: gim_login
      PASSWORD: gim_password

    IWD_DB:
      DSNDEF:
DSN_NAME=IWD_DB;DB_TYPE=POSTGRESQL;DB_TYPE_EX=PostgreSQL;HOST=iwd_db_host;PORT=5432;DB_NAME=dm_gcxi
      LOGIN: iwd_login
      PASSWORD: iwd_password

  deployment:
    deployPostgres: true
    deployLocalPV: true
    useDynamicLogPV: false
```

```
imagePullPolicy:
  worker: IfNotPresent
  control: IfNotPresent

images:
  postgres: postgres:11
```

PVCs required by GCXI

Mount Name	Mount Path (inside container)	Description	Access Type	Default Mount Point on Host (may be changed thru values) These directories MUST pre-exist on your host to accommodate the local provisioner.	Must be Shared across Nodes?	Required Node Label (applies to deafult Local PVs setup)
gcxi-backup	/genesys/gcxi_shared/backup	Backups Used by control container / jobs.	RWX	/genesys/gcxi/backup Can be overwritten by: Values.gcxi.local.pv.backup.path	Not necessarily.	gcxi/local-pv-gcxi-backup = "true"
gcxi-log	/mnt/log	MSTR logs Used by main container. The Chart allows log volumes of legacy hostPath type. This scenario is the default.	RWX	/mnt/log/gcxi subPathExpr: $(POD_NAME) Can be overwritten by: Values.gcxi.local.pv.log.path	Not necessarily.	gcxi/local-pv-gcxi-log = "true" Node label is not required if you are using hostPath volumes for logs.
gcxi-postgres	/var/lib/postgresql/data	Meta DB volume Used by Postgres container, if deployed.	RWO	/genesys/gcxi/shared Can be overwritten by: Values.gcxi.local.pv.postgres.path	Yes, unless you tie PostgreSQL container to a particular node.	gcxi/local-pv-postgres-data = "true"
gcxi-share	/genesys/gcxi_share	MSTR shared caches and cubes.	RWX	/genesys/gcxi/data subPathExpr:	Yes	gcxi/local-pv-gcxi-share =

		Used by main container.		$(POD_NAME) Can be overwritten by: Values.gcxi.local.pv.share.path		"true"

Procedure: 2. Label Nodes

Purpose: Label nodes to allocate PersistentVolumes (PVs). GCXI deployment requires three PVs, plus one for PostgreSQL deployment; see the table PVCs required by GCXI. All four PVs are linked to GCXI pods by means of PersistentVolumeClaims (PVCs). You can create PVs in advance (set **deployLocalPV: false**) or during GCXI installation (set **deployLocalPV: true**). In this example, we allow PVs to be created on all nodes.

Steps

1. Prepare nodes to keep backups — label all worker nodes and create the folder **/genesys/gcxi/backup** on each one:
    1. From the Kubernetes Control plane node, execute the following command for each worker node:
        ```
        kubectl label nodes <<worker node>> gcxi/local-pv-gcxi-backup=true
        ```
    2. On each Kubernetes worker node, execute the following command:
        ```
        mkdir /genesys/gcxi/backup/
        ```
2. Prepare nodes to keep logs — label all worker nodes and create the folder **/mnt/log/gcxi** on each one:
    1. From the Kubernetes Control plane node, execute the following command for each worker node:
        ```
        kubectl label nodes <<worker node>>  gcxi/local-pv-gcxi-log=true
        ```
    2. On each Kubernetes worker node, execute the following command:
        ```
        mkdir /mnt/log/gcxi
        ```
3. Prepare nodes to keep MicroStrategy's cache, cubes, and so on — label all worker nodes, and create the folder **/genesys/gcxi/shared**. This folder must be shared across worker nodes:
    1. From the Kubernetes Control plane node, execute the following command for each worker node:
        ```
        kubectl label nodes <<worker node>>  gcxi/local-pv-gcxi-share=true
        ```
    2. On each Kubernetes worker node, execute the following command:
        ```
        mkdir /genesys/gcxi/shared/
        ```

4. Prepare nodes to keep PostgreSQL database files — either label all worker nodes and create shared folder **/genesys/gcxi/data**, or label one node (to allow PostgreSQL to run only on it) and create the folder on it:
    1. From the Kubernetes Control plane node, execute the following command for the worker node where PostgreSQL will run:

        ```
        kubectl label nodes <<worker node>>  gcxi/local-pv-postgres-data=true
        ```

    2. On the Kubernetes worker node where PostgreSQL will run, execute the following command:

        ```
        mkdir /genesys/gcxi/data
        ```

Procedure: 3. Deploying GCXI

Purpose: Deploy GCXI. This procedure provides steps for environments without LDAP — for environments that include LDAP (or other features not supported in **values.yaml**) you can pass container environment variables such MSTR_WEB_LDAP_ON=true using the **gcxi.envvars** file (for example: `--set-file gcxi.envext=gcxi.envvars`).

Steps

1. For debug purposes, execute the following commands to render templates without installing:

    ```
    helm template --debug -f values-test.yaml gcxi-helm gcxi/
    ```

    Kubernetes descriptors are displayed. The values you see are generated from Helm templates, and based on settings from **values.yaml** and **values-test.yaml**. Ensure that no errors are displayed; you will later apply this configuration to your Kubernetes cluster.

2. To deploy GCXI, execute the following command:

    ```
    helm install --debug --namespace gcxi --create-namespace -f values-test.yaml
    gcxi-helm gcxi/
    ```

    This process takes several minutes. Wait until all objects are created and allocated, and the Kubernetes descriptors applied to the environment appear.

    Genesys recommends that you avoid using helm install with the `--wait` option when deploying GCXI. If you use `--wait`, the Helm architecture post-install hook (which in this case is the gcxi_init job) won't be triggered properly. For more information, see the Helm documentation.

3. To check the installed Helm release, execute the following command:

    ```
    helm list --all-namespaces
    ```

or alternatively, execute the following command:

```
helm list -n gcxi
```

4. To check GCXI Kubernetes objects created by Helm in gcxi release, execute the following command:

```
kubectl get all -n gcxi
```

Procedure: 4. Configuring Ingress

Purpose: To make GCXI accessible from outside the cluster, a special entity called Ingress must be configured. You can find detailed instructions to deploy NGINX Ingress Controller here. The following steps provide an example only.

Steps

1. Log in to the control plane node.
2. Execute the following commands to add the Helm repository:

```
helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
helm repo add stable https://charts.helm.sh/stable
helm repo update
```

3. Execute the following commands to install the NGINX chart:

```
helm install --debug --set controller.kind=DaemonSet --set
controller.hostNetwork=true --set tcp.34952=gcxi/gcxi:mstr --set tcp.8180=gcxi/
gcxi:metrics gcxi-nginx ingress-nginx/ingress-nginx --create-namespace -n
ingress-nginx
```

4. Execute the following commands to verify that the NGINX chart is successfully installed:

```
helm list -n ingress-nginx
```

5. Execute the following commands to verify that NGINX pods are successfully deployed on all cluster nodes:

```
kubectl get pod -n ingress-nginx
```

Maintenance Procedures

This section provides additional procedures, such as troubleshooting steps.

Procedure: Troubleshooting

Purpose: Use the instructions in this section only if you encounter errors or other difficulties. Problems with the deployment are most often associated with the following three kinds of objects:

- PVs
- PVCs
- pods

Steps

1. To list the objects that might cause problems, execute the following commands:

   ```
   kubectl get pv -o wide

   kubectl get pvc -n gcxi -o wide

   kubectl get po -n gcxi -o wide
   ```

2. Examine the output from each **get** command.
3. If any of the objects have a non-ready state (for example, **Unbound** (PVCs only), **Pending**, or **CrashLoop**) execute the following command to inspect the object more closely using **describe**:

   ```
   kubectl describe <type> <name>
   ```

   For example:

   ```
   kubectl describe po gcxi-0
   ```

4. In the **describe** output, inspect the section **Events**.

Procedure: Upgrade GCXI using Helm

Purpose: In scenarios where you have previously deployed Genesys CX Insights using Helm, use the instructions in this procedure to upgrade to a newer Genesys CX Insights release.

Prerequisites

Obtain the Helm installation package (IP) before you begin. The Helm IP has a file name in the

format gcxi-<release>.tgz, for example **gcxi-018.00.tgz**. If SAML is enabled in your environment, before you update GCXI, see the instructions in the *Genesys CX Insights User's Guide* pertaining to updating Genesys CX Insights when SAML is enabled.

Steps

1. On the Control Plane node, set the current directory to the helm folder, and execute the following command to create a subfolder to differentiate this release from others:

   ```
   mkdir helm_<folder>
   ```

   where <folder> is the folder in which to extract the Helm package. For example **018.00**.

2. Execute the following command to extract the Helm package:

   ```
   tar xvzf <helm IP file name> -C helm_<folder>
   ```

   where <folder> is the folder you created in the previous step, and <helm IP file name> is the name of the gcxi Helm package you are installing.

   For example:

   ```
   mkdir helm_018.00; tar xvzf gcxi-9.0.018.00.tgz -C helm_018.00
   ```

3. Execute the following command to remove Ingress:

   ```
   helm delete gcxi-nginx -n ingress-nginx
   ```

4. From the folder where you deployed the previous release of GCXI using Helm, copy the file **values-test.yaml** into the new folder.

5. Navigate to the new folder (for example, using the CD command), and execute the following command:

   ```
   helm upgrade --debug gcxi-helm gcxi/ --namespace gcxi --create-namespace -f
   values-test.yaml
   ```

6. Execute the following command to configure Ingress:

   ```
   helm install --debug --set controller.kind=DaemonSet --set
   controller.hostNetwork=true --set tcp.34952=gcxi/gcxi:mstr --set tcp.8180=gcxi/
   gcxi:metrics gcxi-nginx ingress-nginx/ingress-nginx --create-namespace -n
   ingress-nginx
   ```

Procedure: Uninstall GCXI

Purpose: To remove GCXI

Steps

1. Execute the following command to remove Ingress:

   ```
   helm delete gcxi-nginx -n ingress-nginx
   ```

2. Execute the following command to remove GCXI:

   ```
   helm uninstall gcxi-helm -n gcxi
   ```

Keep in mind that, after deploying Genesys CX Insights, you must perform additional post-installation setup steps before actively using the reports and projects. After completing the deployment steps on this page, complete the following:

- Installing report editing software
- Post-Installation steps

Installing Genesys CX Insights - OpenShift using Helm

Deploy Genesys CX Insights using Red Hat OpenShift. This deployment method is suitable for production environments; for other deployment options, see Choose a deployment type and Prerequisites.

This is an example scenario — This page provides a high-level outline, illustrating one scenario to help you visualize the overall process. Genesys does not provide support for OpenShift or other third-party products, so you must have knowledge of OpenShift and other products to complete this type of installation. This page describes an example of deployment with OpenShift Cluster 4.5.16.

> ### Important
> Beginning with release 9.0.016, Genesys CX Insights supports deployment using Red Hat OpenShift. This deployment option is available as part of Genesys Engage cloud private edition Early Adopter Program.
>
> *Please note:* Until Genesys further expands our support of OpenShift on the Genesys Engage cloud private edition platform, Genesys CX Insights supports the basic installation of containers on customer-operated OpenShift clusters. Customers own the responsibility of deploying and maintaining OpenShift clusters, and Genesys provides support only for issues related to GCXI containers.

Prerequisites for deploying using OpenShift

Before you begin, ensure that:

- Your OpenShift cluster is up and running, with nodes in the **Ready** state.
- For HA deployments, at least two worker nodes are available for GCXI pods.
- Each Worker machine meets the following minimum requirements:
  - 64-bit compatible CPU architecture. (2 or more CPUs).
  - 10 GB for each GCXI container, and 2 GB for the PostgreSQL container. Production deployments commonly reserve 16 – 64 GB RAM for each container.
  - 40 GB of available disk space if loading images from a repository
- OpenShift client and Helm-3 are installed on the host where the deployment will run.
- Images **gcxi** and **gcxi_control** are properly tagged and loaded on the registry. OpenShift will pull the images from there to each OpenShift worker node during deployment.
- On each worker node, values are set for `kernel.sem` and `vm.max_map_count`, as required by

MicroStrategy. For example:

```
echo "kernel.sem = 250 1024000 250 4096" >> /etc/sysctl.conf
echo "vm.max_map_count = 5242880" >> /etc/sysctl.conf
sysctl -p
```

PVCs required by GCXI

Mount Name	Mount Path (inside container)	Description	Access Type	Default Mount Point on Host (can be changed through values; these directories MUST pre-exist on your host to accommodate the local provisioner)	Shared across Nodes?	Required Node Label (applies to deafult Local PVs setup)
gcxi-backup	/genesys/gcxi_shared/backup	Backups Used by control container / jobs.	RWX	/genesys/gcxi/backup Can be overwritten by: Values.gcxi.local.pv.backup.path	Not necessarily.	gcxi/local-pv-gcxi-backup = "true"
gcxi-log	/mnt/log	MSTR logs Used by main container. The Chart allows log volumes of legacy hostPath type. This scenario is the default.	RWX	/mnt/log/gcxi subPathExpr: $(POD_NAME) Can be overwritten by: Values.gcxi.local.pv.log.path	Must not be shared across nodes.	gcxi/local-pv-gcxi-log = "true" Node label is not required if you are using hostPath volumes for logs.
gcxi-postgres	/var/lib/postgresql/data	Meta DB volume Used by Postgres container, if deployed.	RWO	/genesys/gcxi/shared Can be overwritten by: Values.gcxi.local.pv.postgres.path	Yes, unless you tie PostgreSQL container to a particular node.	gcxi/local-pv-postgres-data = "true"
gcxi-share	/genesys/gcxi_share	MSTR shared caches and cubes. Used by main container.	RWX	/genesys/gcxi/data subPathExpr: $(POD_NAME) Can be overwritten by: Values.gcxi.local.pv.share.path	Yes	gcxi/local-pv-gcxi-share = "true"

Deploying GCXI with OpenShift

The following procedures describe example steps to deploy GCXI with OpenShift. The exact steps required will vary for your environment.

Procedure: 1. Preconfigure the environment

Purpose: Ensure that the environment is properly prepared for deployment.

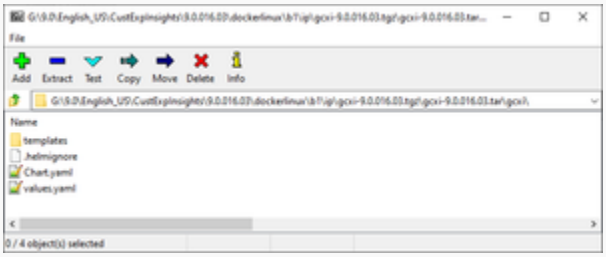
Steps

1. Ensure that the GCXI project has been created.
2. Ensure that four PersistentVolumes (PV) have been created. See the table PVCs required by GCXI.
3. Ensure that security context constraints (SCC) in OpenShift are configured appropriately to allow OpenShift Container Platform to run containers using user ID = 500. For more information, see About security context constraints in the Open Shift documentation.
   For test and development environments, execute the following command if you wish to run pods as any user:

   ```
   oc adm policy add-scc-to-user anyuid -z default
   ```

Procedure: 2. Prepare for deployment

Purpose: Prepare the environment, and gather files needed for deployment.



Contents of the Helm IP

Prerequisites

Within the the Genesys Customer Experience Insights package for the Docker Linux OS, look for the Helm installation package (IP) — a small TGZ file (for example **gcxi-9.0.018.00.tgz**)

that contains the Helm files. You require these files to complete this procedure.

Steps

1. On the host where the deployment will run, create a folder: **helm**.
2. Copy the Helm installation package (for example **gcxi-9.0.018.00.tgz**) into the **helm** folder, and extract the archive into a subfolder called **helm/gcxi**.
3. View the file **helm/gcxi/Chart.yaml**, and ensure that the appVersion is set to the desired GCXI version.
4. Open the file **helm/gcxi/values.yaml**, and follow the instructions it provides to guide you in creating a new file, **values-test.yaml** with appropriate settings. Save the new file in the **helm** folder.
   For example, the following content in the **values-test.yaml** file is appropriate for a simple deployment using PostgreSQL inside the container, with PersistentVolumes named **gcxi-log-pv**, **gcxi-backup-pv**, **gcxi-share-pv**, and **gcxi-postgres-pv** (which are deployed in Step 2 of Procedure: 1. Preconfigure the environment). Create content in the **values-test.yaml** file that is appropriate for your environment:

```
gcxi:
  env:
    GCXI_GIM_DB:
      DSNDEF:
DSN_NAME=GCXI_GIM_DB;DB_TYPE=POSTGRESQL;DB_TYPE_EX=PostgreSQL;HOST=gim_db_host;PORT=5432;DB_NAME=gi
      LOGIN: gim_login
      PASSWORD: gim_password

    IWD_DB:
      DSNDEF:
DSN_NAME=IWD_DB;DB_TYPE=POSTGRESQL;DB_TYPE_EX=PostgreSQL;HOST=iwd_db_host;PORT=5432;DB_NAME=dm_gcxi
      LOGIN: iwd_login
      PASSWORD: iwd_password

    PGDATA: /var/lib/postgresql/data/mydata4

  deployment:
    deployPostgres: true
    deployLocalPV: false
    useDynamicLogPV: false
    useHostPathLogInitContainer: true
    hostIPC: false

  imagePullPolicy:
    worker: IfNotPresent
    control: IfNotPresent

  replicas:
    worker: 2

  images:
    postgres: postgres:11

  pvc:
    log:
```

```
  volumeName: gcxi-log-pv
backup:
  volumeName: gcxi-backup-pv
share:
  volumeName: gcxi-share-pv
postgres:
  volumeName: gcxi-postgres-pv
```

Procedure: 3. Deploy GCXI

Purpose: Deploy GCXI. This procedure provides steps for environments without LDAP — for environments that include LDAP (or other features not supported in **values.yaml**) you can pass container environment variables such MSTR_WEB_LDAP_ON=true using the **gcxi.envvars** file (for example: `--set-file gcxi.envext=gcxi.envvars`).

Steps

1. Log in to OpenShift cluster from the host where you will run deployment; for example, by executing the following command:

   ```
   oc login --token <token> --server <url of api server>
   ```

2. Execute the following command to make the GCXI project the default:

   ```
   oc project gcxi
   ```

3. For debug purposes, execute the following command to render templates without installing:

   ```
   helm template --debug -f values-test.yaml gcxi-helm gcxi/
   ```

   Kubernetes descriptors are displayed. The values you see are generated from Helm templates, and based on settings from **values.yaml** and **values-test.yaml**. Ensure that no errors are displayed; you will later apply this configuration to your Kubernetes cluster.

4. To deploy GCXI, execute the following command:

   ```
   helm install --debug --namespace gcxi --create-namespace -f values-test.yaml
   gcxi-oc gcxi/
   ```

   This process takes several minutes. Wait until all objects are created and allocated, and the Kubernetes descriptors applied to the environment appear.

5. To check the installed Helm release, execute the following command:

   ```
   helm list –all-namespaces
   ```

6. To check the GCXI project status, execute the following command:

```
oc status
```

7. To check GCXI OpenShift objects created by Helm, execute the following command:

   ```
   oc get all -n gcxi
   ```

8. Make GCXI accessible from outside the cluster, using the standard HTTP port. For production environments, Genesys recommends that you create secure routes as discussed on the OpenShift website. For testing or development environments, perform the following steps:

   1. Execute the following command to expose the gcxi service:

      ```
      oc expose service gcxi --port web --name web
      ```

   2. Execute the following command to verify that the new route is created in the gcxi project:

      ```
      oc get route -n gcxi
      ```

      Route information appears, similar to the following:

      ```
      NAME      HOST/PORT                              PATH   SERVICES   PORT
      TERMINATION   WILDCARD
      web       web-gcxi.<host>                               gcxi
      web                      None
      ```

      where <host> is the host name generated by OpenShift.

9. Verify that you can now access GCXI at the following URL:

   ```
   http://web-gcxi.<host>/MicroStrategy/servlet/mstrWeb
   ```

Procedure: 4. Install a new License key

Purpose: The MicroStrategy server instance that runs in the container includes a temporary pre-activated key, which is required for the operation of MicroStrategy. Use the steps in this procedure to install a new license key by setting the MSTR_LICENSE variable.

Prerequisites

Obtain a new license key; contact your Genesys Customer Care representative for assistance.

Steps

1. Execute the following command to back up the GCXI meta db:

```
kubectl apply -f <destination path>/gcxi-backup.yaml
```

where <destination path> is the folder in which the Genesys IP is stored, for example:

```
kubectl apply -f /genesys/gcxi/gcxi-backup.yaml
```

2. Execute the following commands to stop currently running containers:

```
kubectl scale deploy/gcxi-secondary --replicas=0
```

```
kubectl scale deploy/gcxi-primary --replicas=0
```

3. Open your **values-test.yaml** file for editing (or open the file **helm/gcxi/values.yaml**, and follow the instructions it provides to guide you in creating a new file, **values-test.yaml** with appropriate settings.)
4. Populate the MSTR_LICENSE environment variable, and save the new file in the **helm** folder.
5. Execute the helm upgrade command to restart the pods:

```
helm upgrade --debug gcxi-helm gcxi/ --namespace gcxi --create-namespace -f
 values-test.yaml --recreate-pods
```

Maintenance Procedures

This section provides additional procedures, such as troubleshooting steps.

Procedure: Troubleshooting

Purpose: Use the instructions in this section only if you encounter errors or other difficulties. Problems with the deployment are most often associated with the following three kinds of objects:

- PVs
- PVCs
- pods

Steps

1. To list the objects that might cause problems, execute the following commands:

```
oc get pv -o wide

oc get pvc -o wide -n gcxi

oc get po -o wide -n gcxi
```

2. Examine the output from each **get** command.
3. If any of the objects have a non-ready state (for example, **Unbound** (PVCs only), **Pending**, or **CrashLoop**) execute the following command to inspect the object more closely using **oc describe**:

   ```
   oc describe <type> <name>
   ```

   For example:

   ```
   oc describe po gcxi-0
   ```

4. In the **describe** output, inspect the section **Events**.

Procedure: Uninstall GCXI

Purpose: To remove GCXI

Steps

1. To remove GCXI, execute the following command:

   ```
   helm uninstall gcxi-oc -n gcxi
   ```

Keep in mind that you must perform additional post-installation setup steps before actively using the reports and projects. After completing the steps on this page, complete the following:

- Installing report editing software
- Post-Installation steps

Installing Genesys CX Insights - Docker Compose

This page describes a simplified procedure that you can optionally use to deploy Genesys CX Insights for demonstration, testing, evaluation, or development purposes; when deployed on CentOS, this method is also a suitable choice for small to medium production environments. This page describes deploying Genesys CX Insights and supporting software on Windows or CentOS using a single docker-compose file on a single virtual machine (VM).

For most production environments, Genesys recommends deploying Docker with Kubernetes or OpenShift; see Choose a deployment type and Before you install Genesys CX Insights. It is possible to install using other configurations; refer to the Docker website for information about other scenarios.

Before You Begin

Genesys CX Insights requires a suitably-prepared environment in order to operate successfully, including properly-configured installations of each of the following:

- A supported release of Microsoft Windows or CentOS. For information about what releases are supported, see the Docker website: Microsoft Windows / CentOS.
- If you plan to link Genesys CX Insights to your Info Mart, install Genesys Info Mart and RAA:
  - Genesys Info Mart release 8.5 database — The Genesys Info Mart documentation set describes how to deploy and configure Genesys Info Mart, including information about hardware sizing requirements to support Gensys Info Mart. Genesys CX Insights can provide meaningful reports only if the Info Mart database is regularly populated by a Genesys Info Mart 8.5 application. Genesys Info Mart must be properly configured and installed before Genesys CX Insights runs the aggregation process (RAA). Refer to the *Genesys Info Mart Deployment Guide* or the *Genesys Migration Guide* for information that pertains to configuring, installing, or upgrading Genesys Info Mart.
  - Reporting and Analytics Aggregates (RAA) — The RAA documentation set describes how to deploy RAA, and how to configure the aggregation process.
- Ensure that you have the latest Genesys CX Insights 9.0 installation packages (IP); talk to your Genesys representative for information about where to download the installation packages.

Installation packages for GCXI

Component	IP / File	tar files*
CustExInsights — Genesys Customer Experience Insights	Docker container (Docker Linux platform) **IP_CustExpInsights_9000XXX_ENU_dockerlinux.zip**	**gcxi.tar.gz** — contains the **gcxi** Docker image, which contains a fully installed Microstrategy

	(where XXX is the latest release number.)	Server 10.x (the latest supported release of MicroStrategy: 11.2.1, for example.). This container provides a *stateless* deployment, where project data (reports, users, and other objects) is stored separately in a MicroStrategy *meta* database. This image is used for production deployments.
	Regular Linux IP (Linux platform) **IP_CustExpInsights_9000XXX_ENU_linux.tar.gz** (where XXX is the latest release number.)	**data.tar.gz** — contains the various YAML files (Kubernetes script files), such as gcxi.yaml, gcxi-postgres-yaml, and gcxi-init.yaml, and the gcxi.properties file (files which you must edit as part of the deployment procedure), PostgreSQL database dump with MicroStrategy meta-data database for GCXI project, and other files needed for GCXI
CustExInsightOps — Genesys Customer Experience Insights Ops	Docker container (Docker Linux platform) **IP_CustExpInsightsOPS_9000XXX_ENU_dockerlinux.zip** (where XXX is the latest release number.)	**gcxi_control.tar.gz** — contains the **gcxi_control** Docker image, which is used for deployment and configuration of the GCXI solution.
CustExInsightDB — Genesys Customer Experience Insights DB (Discontinued beginning with GCXI release 9.0.019.01)	Docker container (Docker Linux platform) **IP_CustExpInsightsDB_9000XXX_ENU_dockerlinux.zip** (where XXX is the latest release number.)	**gcxi_postgres.tar.gz** — contains the **gcxi_postgres** image, which contains a PostgreSQL database server with GCXI MicroStrategy Project, Meta data, and History databases pre-deployed. This image is discontinued beginning with GCXI release 9.0.019.01
MSSecEntPltf64 — MicroStrategy Secure Enterprise Platform for Windows	MicroStrategy software for Windows (server and client / editing tools) **MicroStrategy_XXX_IntelligentEnterprise_Windows_XXX.zip** (where XX is the current MicroStrategy release. For example, **MicroStrategy_11.3_IntelligentEnterprise_Windows_11.3.0560.0066.zip**.)	MicroStrategy_Secure_Enterprise_Platform_11_3_Win/cpe1705/PI
MSWrkstn — MicroStrategy Workstation	MicroStrategy Workstation software for Windows **workstation-win-ent_XXX.zip** (where XXX is the current MicroStrategy release. For example, **workstation-win-**	MicroStrategy_Workstation_11.3.63/ The MicroStrategy Workstation package is available beginning with GCXI release 100.0.029.0000

	ent_11.3.63.zip.)	

*Important:

In some releases, the names of the container images in the installation package differ from the description in the **Installation packages for GCXI** table. In these scenarios, rename the container images as described in the table **Renaming the images**:

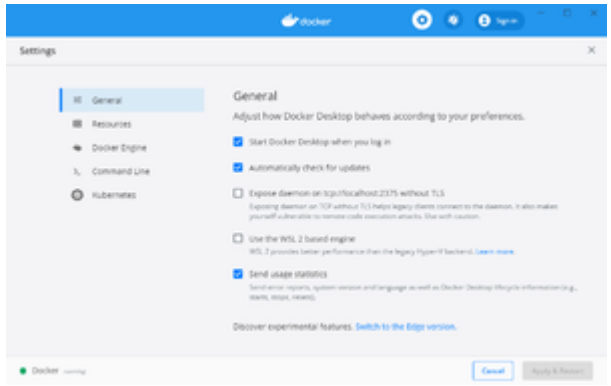
Renaming the images (if your release requires it)

Copy this file from this folder to a convenient location on your local hard drive (for example `C:\GCXI_temp`):	Rename it as:
CustExpInsights ... **dockerlinux**... **9.0.010.04.tar.gz**	**gcxi.tar.gz**
CustExpInsightsDB ... **9.0.010.04.tar.gz** (This image is discontinued beginning with GCXI release 9.0.019.01)	**gcxi_postgres.tar.gz**
CustExpInsightsOps ... **9.0.010.04.tar.gz**	**gcxi_control.tar.gz**

Note that Reporting and Analytics Aggregates (RAA) files are also available in the same location (**Reporting_and_Analytics_Aggregates_G231_850XXXX_ENU_ISO**). See the Reporting and Analytics Aggregates documentation for more information about deploying RAA.

Installing Docker

Docker Compose deployments have the same general requirements as Kubernetes deployments, with two key differences: Only one VM is required, and instead of deploying Docker and Kubernetes on Linux, you deploy only Docker, and you do so on either Linux or Windows.



Docker Desktop

Procedure: 1. Install Docker

Purpose: This procedure provides an example deployment procedure for installing Docker, without Kubernetes. This is suitable for demo, test, or development environments, not for production.

Prerequisites

- If you have previous Docker images installed, optionally back them up, as the steps in this procedure will remove them.

Steps

Installing Docker for Windows

Before you begin, ensure that your environment meets the minimum hardware requirements for Docker for Windows. At least 16 GB of RAM is

required, of which 12 GB should be available for Docker (which provides 4 GB for Hyper-V, 10 GB for the GCXI container, and 2 GB for the PostgreSQL container). More is recommended, particularly if you plan to use this deployment as a production environment.

1. Download the latest stable release of Docker Desktop, from the Docker website.
2. Open **Docker Desktop Installer** and follow the Docker Desktop for Windows instructions. When prompted during the installation process, clear the "Enable WSL2 Windows Features" option.
3. Using **Run as administrator**, start **Docker Desktop**.
4. Choose **Settings** > **General**, and disable **Use the WSL 2 based engine**.
5. Click **Apply and Restart**.

OR

Installing Docker for CentOS

Before you begin, ensure that your environment meets the OS requirements. At least 12 GB of RAM is required (10 GB for the GCXI container, and 2 GB for the PostgreSQL container). More is recommended, particularly if you plan to use this deployment as a production environment.

1. Execute the following command to verify that the **centos-extras** repository is enabled:

   ```
   sudo yum repolist
   ```

   The **centos-extras** repository is enabled by default.
2. Execute the following command to uninstall old versions:

   ```
   sudo yum remove docker \
                     docker-client \
                     docker-client-latest \
                     docker-common \
                     docker-latest \
                     docker-latest-logrotate \
   ```

```
                  docker-logrotate \
                  docker-engine
```

3. Execute the following command to install the yum-utils package and set up the stable repository:

   ```
   sudo yum install -y yum-utils
    sudo yum-config-manager \
        --add-repo \
        https://download.docker.com/linux/centos/docker-ce.repo
   ```

4. Execute the following command to install the Docker engine:

   ```
   sudo yum install docker-ce docker-ce-cli containerd.io
   ```

5. Execute the following command to start Docker:

   ```
   sudo systemctl start docker
   ```

6. Execute the following command to verify that the engine is installed and running:

   ```
   sudo systemctl status docker
   ```

7. Create a group and user, to simplify management:
   1. Execute the following command to create the group 'docker': `sudo groupadd docker`
   2. Execute the following command to add a user to the group: `sudo usermod -aG docker $USER`
      where USER is the user name for the account you will use to install and manage Docker.
   3. Execute the following command to activate the change to the group: `newgrp docker`
   4. Log out, and log in using the USER account you added to the 'docker' group.
   5. Execute the following command to verify that you can now run docker commands without using sudo: `docker run hello-world`
      A container runs, displays the 'hello world' message, and exits.

   For more information, and other options for installing Docker for CentOS, see Install Docker Engine on CentOS on the Docker web site.

Procedure: 2. Cleaning up your Docker VM

Purpose: If you have previously installed Genesys CX Insights and supporting software or are preparing to reinstall or update it, use the instructions in this procedure to clean up your Docker Virtual Machine (VM).

> ### Important
> This procedure deletes all Genesys CX Insights content, including customizations you may have made to the Genesys CX Insights Project or reports.

Steps

1. If it's not already running, start Docker (On CentOS, log in using an account in the 'docker' user group, and run `systemctl start docker`. On Windows, use **Run as administrator** to start **Docker Desktop**), and open a command terminal, such as PowerShell.
2. Execute the following command to see what containers are running:

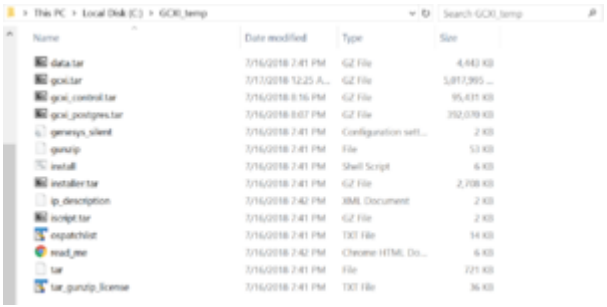
   ```
   docker ps -a
   ```

3. If any containers are running, make note of the container IDs or names, and execute the following command to remove them:

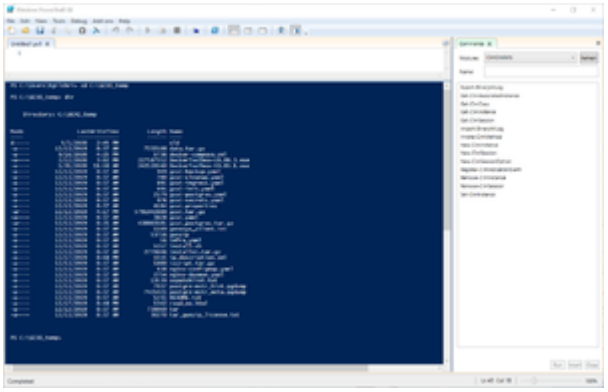
   ```
   docker rm -f <container ID or name>
   ```

4. Execute the following command to remove all existing containers, images and volumes:

   ```
   docker system prune -af --volumes
   ```



The installation package contents



GCXI_temp

Procedure: 3. Loading Docker images

Purpose: This procedure describes the steps you take to load the Genesys CX Insights installation files into Docker.

Prerequisites

- You must have available the Genesys CX Insights 9.0 installation package (IP), which includes the Genesys CX Insights repository file, containing the files listed in the table **The Genesys CX Insights installation package**; talk to your Genesys representative for information about where to download the installation packages. Genesys recommends that you save these files into a folder near the root, to make the path easier to type; for example, `C:\GCXI_temp`.

The Genesys CX Insights installation package

File	Description
gcxi.tar.gz container image	Contains a fully installed Microstrategy Server 10.x (the latest release of MicroStrategy: 10.11, for example.), providing a *stateless* deployment, where project data (reports, users, other objects) is stored separately in a MicroStrategy *meta* database. This option requires that you have configured an external PostgreSQL server.
gcxi_postgres.tar.gz container image	Contains a Genesys CX Insights Project with a PostgreSQL database server running with GCXI meta / history databases deployed; so it includes reports, users, and other objects in a single container.
data.tar.gz	Assorted deployment descriptor (YAML) files, including: • **docker-compose.yml** — A file that you use to start the solution, and that you optionally edit in some procedures.
gcxi_control.tar.gz	Contains the **gcxi_control** Docker image, which is used for deployment and configuration of the GCXI solution.

- In Release 9.0.010.04 and later, the names of the container images in the installation package differ from the description in the **The Genesys CX Insights installation package** table. If necessary, rename the container images so they match the names given in the following table:

Renaming the images in 9.0.010.04 and later

Copy the file from this folder to a convenient location on your local hard drive (for example `C:\GCXI_temp`):	Rename it as:
CustExpInsights ... **dockerlinux**... **9.0.015.01.tar.gz**	**gcxi.tar.gz**
CustExpInsightsDB ... **9.0.015.01.tar.gz**	**gcxi_postgres.tar.gz**

Steps

1. If it's not already running, start Docker (On CentOS, log in using an account in the 'docker' user group, and run `systemctl start docker`. On Windows, use **Run as administrator** to start **Docker Desktop**), and open a command terminal, such as PowerShell.
2. Change the current directory to the folder where you saved the Genesys CX Insights installation files. For example:

   ```
   cd C:\GCXI_temp
   ```

3. In release 9.0.019.00 and earlier, execute the following commands to load the PostgreSQL Docker image:

   ```
   docker load -i gcxi_postgres.tar.gz
   ```

4. Execute the following commands to load the Docker images:

   ```
   docker load -i gcxi.tar.gz
   ```

5. Execute the following command to verify that the images loaded correctly:

   ```
   docker images
   ```

   The console lists the installed Docker images:

```
$ docker images
REPOSITORY                                                TAG                 IMAGE ID            CREATED             SIZE
pureengage-docker-production.jfrog.io/gcxi/gcxi           9.0.015.01          e7a7216f2f2f        4 months ago        11.7GB
pureengage-docker-production.jfrog.io/gcxi/gcxi_postgres  9.0.015.01          068b8c6ba06c        4 months ago        3.53GB
```

6. Execute the following commands to retag the images (note that the PostgreSQL image is not used in release 9.0.019.01 and later):

```
docker tag <REPOSITORY>/gcxi:<RELEASE> gcxi

docker tag <REPOSITORY>/gcxi_postgres:<RELEASE> gcxi_postgres
```

   where:
   <REPOSITORY> is the full repository path shown in the preceding step (such as pureengage-docker-production.jfrog.io/gcxi/). The repository path varies depending on the release, and is not present or required in some releases.

   <RELEASE> is the a string corresponding to the release you are installing (such as 9.0.015.01),

   For example:

```
docker tag pureengage-docker-production.jfrog.io/gcxi/gcxi:9.0.019.01 gcxi

docker tag pureengage-docker-production.jfrog.io/gcxi/gcxi_postgres:9.0.019.01 gcxi_postgres
```

7. Execute the following command to verify that the images loaded correctly, and have correct tagging:

```
docker images
```

   The console lists the Docker images:

```
$ docker images
REPOSITORY                                                TAG                 IMAGE ID            CREATED             SIZE
gcxi                                                      latest              e7a7216f2f2f        4 months ago        11.7GB
pureengage-docker-production.jfrog.io/gcxi/gcxi           9.0.015.01          e7a7216f2f2f        4 months ago        11.7GB
gcxi_postgres                                             latest              068b8c6ba06c        4 months ago        3.53GB
pureengage-docker-production.jfrog.io/gcxi/gcxi_postgres  9.0.015.01          068b8c6ba06c        4 months ago        3.53GB
```

   Compare the result to the figure; each image must have a name in the REPOSITORY column with no preceding path, and a value of LATEST in the TAG column. Note that each image appears twice in the list; this is expected behavior, because each one has two tags.

> **Important**
>
> The MicroStrategy server instance that runs in the container includes a pre-activated key, which is required for the operation of MicroStrategy. The key expires on the last day of each year; when this happens, download the latest release of the Genesys CX Insights installation package, and restart your containers using the new image.

Procedure: 4. Specify a database

Purpose: Tell Genesys CX Insights what database to use — either the included sample / demo database, or your external Genesys Info Mart database.

Steps

Choose one the of the following methods:

Use the provided sample / demo database

This method uses the images **gcxi** and **gcxi_postgres**.

1. Open the (CustExpInsights\linux\b1\ip\) **data.tar.gz** package, and copy the file **docker-compose.yml** into the folder where you stored the installation package, for example **C:\GCXI_temp**.
2. Open the **docker-compose.yml** file for editing.

3. In the **services:** section, below the line that begins *old version of gcxi-postgres service*, uncomment the following lines:

```
# gcxi-postgres:
   #   image: gcxi_postgres:9.0.015.00
   #   hostname: gcxi-postgres
   #   volumes:
   #   - "gcxi_postgres:/var/lib/postgresql/data"
   #   ports:
   #   - "5432:5432"
   #   networks:
   #   - gcxi
```

   On Windows deployments, the port 8080 is sometimes used by another process. In this scenario, change the port mapping; Edit the line **8080:8080**, changing the first value to **<unused_ port_in_windows>:8080", where <unused_ port_in_windows> is an unused port, for example** 8280:8080**. If you remap this port, be sure to use the new port value when accessing MicroStrategy web interface.**

4. Comment out from the line that begins *new version of gcxi-postgres service*, down to the end of the **gcxi-postgres:** and **gcxi-control:** sections. Leave **gcxi-0:** and subsequent sections uncommented.
5. If you are connecting the GCXI Docker Compose deployment to a demo Info Mart database that resides in a PostgreSQL container, comment out the DSNDEF lines.
6. Save the **docker-compose.yml** file.

OR

Connect to your Genesys Info Mart database

You must have available all relevant Genesys Info Mart information, including the RDBMS type (Microsoft SQL Server, PostgreSQL, Oracle), hostname, and user credentials. This method uses the images **gcxi** and **gcxi_control**.

1. Change the current directory to the folder where you saved the Genesys CX Insights installation files. For example:

```
cd C:\GCXI_temp
```

2. Execute the following command to load the gcxi_control Docker image:

```
docker load -i gcxi_control.tar.gz
```

3. Execute the following command to retag the image:

```
docker tag <REPOSITORY>/gcxi_control:<RELEASE> gcxi_control
```

where:
<REPOSITORY> is the full repository path shown in the preceding step (such as pureengage-docker-production.jfrog.io/gcxi/). The repository path varies depending on the release, and is not present or required in some releases.

<RELEASE> is the a string corresponding to the release you are installing (such as 9.0.015.01),

For example:

```
docker tag pureengage-docker-production.jfrog.io/gcxi/gcxi_control:9.0.015.01 gcxi_control
```

4. Open the (CustExpInsights\linux\b1\ip\) **data.tar.gz** package, and copy the file **docker-compose.yml** into the folder where you stored the installation package, for example **C:\GCXI_temp**.
5. Open the *docker-compose.yml* file for editing.
6. Some lines in the file are commented out with a single #. Uncomment lines with **DSNDEF_** variables, or those with **GIM_** variables as appropriate, and populate them with suitable values. Note that values of Database type (DB_TYPE) and Database Type extended (DB_TYPE_EX) are allowed, as described in the **docker-compose.yml** file. For example, if you plan to use your own MSSQL Info Mart database, instead of the built-in that comes as part of the container, starting with the line that begins *environment*, uncomment the lines that have a single #:

```
## DSNDEF* is a new DSN definition format suitable for GCXI v. >= 9.0.010.00
 ## If at least one DSNDEF* variable is defined, GIM_* variables are ignored
 ## Each DSNDEF node represents one DSN definition
 ## As of GCXI v. 9.0.010.00 DSN_NAMEs must be predefined:
 ## GCXI_GIM_DB = for project GCXI / CX Insights
 ## IWD_DB = for project IWD
 ## DSNs defined with other names will be created in MSTR, but not used by default
 ## Password notice: if GIM password contains semicolon, it must be escaped with \
```

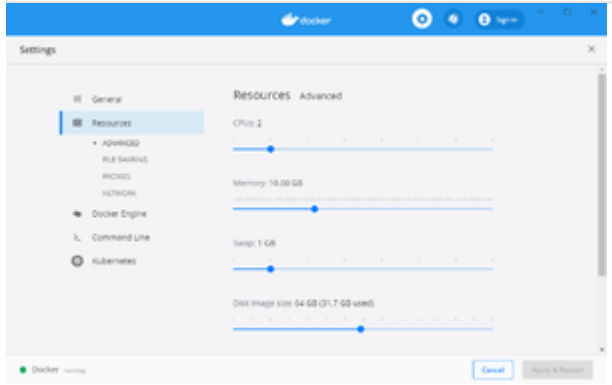
```
## Eg: PASSWORD=my;passwd;.. => PASSWORD=my\;passwd;..
## NB: password notice does not apply to old GIM_* DSN format, no need to escape anything there
## For DB_TYPE and DB_TYPE_EX values see explanation below
          - DSNDEF1=DSN_NAME=GCXI_GIM_DB;DB_TYPE=SQLSERVER;DB_TYPE_EX=Microsoft SQL Server
2012;HOST=gi2-qadb;PORT=1433;DB_NAME=gim85test2voice;LOGIN=gim85test2voice;PASSWORD=gim85test2voice
          - DSNDEF2=DSN_NAME=SOME_NAME;DB_TYPE=SQLSERVER;DB_TYPE_EX=Microsoft SQL Server
2012;HOST=gi2-qadb;PORT=1433;DB_NAME=gim85test2mm;LOGIN=gim85test2mm;PASSWORD=gim85test2mm
          -
DSNDEF3=DSN_NAME=IWD_DB;DB_TYPE=POSTGRESQL;DB_TYPE_EX=PostgreSQL;HOST=gi2-cent7-2;PORT=5435;DB_NAME=gim;LOGIN=gim_db;PASSWORD=gim_db
## Legacy GIM_* syntax - for v. < 9.0.010.00
## Database Type: values allowed: SQLSERVER POSTGRESQL ORCLW
          # - GIM_DB_TYPE=POSTGRESQL
## Database Type extended  (must correspond Type above), values allowed:
## 'Microsoft SQL Server 2012' 'Microsoft SQL Server 2014' 'Microsoft SQL Server 2016'
## 'PostgreSQL'
## 'Oracle 12cR2'  'Oracle 18c'  'Oracle 19c'
## sometimes these types change with new MSTR release
## if values above don't work (e. g. outdated), refer to file '$MSTR_INSTALL_HOME/install/DATABASE.PDS'
## in this file MSTR keeps DB type aliases for the current release
## search for 'DSSOBJECT' element, 'NAME' attribute
## another way: try to create DB Connection in MSTR Developer, and refer to the list of values it suggests
          # - GIM_DB_TYPE_EX=PostgreSQL
          # - GIM_HOST=gi2-qadb
          # - GIM_PORT=5432
## For Postgre and MS SQL this is GIM database name, for Oracle this is not set
          # - GIM_DB=gim85test2voice
## GIM Oracle SID - for GIM Oracle only (set either SID or Service name)
          # - GIM_ORCL_SID=
## GIM Oracle Service name - for GIM Oracle only (set either SID or Service name)
          # - GIM_ORCL_SNAME=
          # - GIM_LOGIN=gim85test2voice
          # - GIM_PASSWORD=gim85test2voice
```

7. Save the **docker-compose.yml** file.

Tip

- When starting the container, if you encounter an error about "exited with status 1", verify that all variables listed above are correctly populated.
- If you are connecting the GCXI Docker Compose deployment to a demo Info Mart database that resides in a PostgreSQL container, comment out the DSNDEF lines.
- If you are connecting to a Named Instance on a Microsoft SQL Server DBMS using DSNDEF environment variables, add 4 backslashes between the hostname and the instance name:

```
HOST=<serverName>\\\\<InstanceName>
```



Docker Desktop Memory Settings

Procedure: 5. Configure memory settings (Windows deployments)

Purpose: This procedure describes the steps you take to configure the virtual environment on Windows deployments.

Steps

1. Open Hyper-V Manager, and change the memory settings for the virtual machine:
    1. Click **Turn Off** to stop the virtual machine.
    2. Open the **Settings** of your virtual machine, and in the section **Memory** > **Hardware**, mark the checkbox **Enable Dynamic Memory**, and click **OK**..
    3. Click **Start** to start the virtual machine.
    4. Close Hyper-V Manager.
2. Open Docker Desktop, and change the memory settings for Docker:
    1. Click **Settings** > **Resources**.
    2. Increase the value of **Memory** to 10GB or more.
    3. Click **Apply & Restart**.

Procedure: 6. Starting Genesys CX Insights containers

Purpose: This procedure describes the steps you take to start, stop, or reset the containers.

Steps

1. If you haven't already done so, open the (CustExpInsights\linux\b1\ip\) **data.tar.gz** package, and copy the file **docker-compose.yml** into the folder where you stored the installation package, for example **C:\GCXI_temp**.
2. If it's not already running, start Docker (On CentOS, log in using an account in the 'docker' user group, and run `systemctl start docker`. On Windows, use **Run as administrator** to start **Docker Desktop**), and open a command terminal, such as PowerShell.
3. Change the current directory to the location where the Genesys CX Insights installation package is stored, for example:

   ```
   cd C:\GCXI_temp
   ```

4. To start the whole solution, enter the following command:

   ```
   docker-compose -f docker-compose.yml up
   ```

   After several minutes (as few as two, but sometimes more than ten, depending on your environment), summary information appears in the console:

   ```
   gcxi-0_1        | Attempt to start PDF Export Service...
   gcxi-0_1        | PDF Export Service is started.
   ...
   gcxi-0_1        | Tomcat started.
   gcxi-0_1        | USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
   gcxi-0_1        | root      1459  3.0  0.0  51760  3864 ?        R    16:33   0:00 ps -auxw --sort pmem
   gcxi-0_1        | root         1  0.0  0.0  12932  4032 ?        Ss   16:32   0:00 /bin/bash /genesys/gcxi/mstr_start.sh
   ...
   ```

   [followed by several more lines]

   > **Tip**

> After docker-compose.yml completes, you will not see a blinking insertion point in the terminal. This is expected behavior. Once you see the output shown above, which may be followed by additional lines, you can open another terminal and proceed to the next step Accessing MicroStrategy web interface.

The containers are now loaded and ready to use.

Procedure: 7. Accessing MicroStrategy web interface

Purpose: Use the following procedure to access the various web interfaces, where you can view the reports and dashboards or manage the software.

Prerequisites

If you remapped the 8080 port in **Procedure: 4. Specify a database**, be sure to use the new port value in this procedure, instead of 8080.

Steps

1. Open a command terminal, such as PowerShell.
2. Execute the following command to learn the IP address of the VM:
   Windows:

   ```
   ipconfig
   ```

CentOS:

```
ip addr show
```

Mac:

```
localhost
```

3. To access the various web interfaces, enter the following addresses (where <VM IP> is the IP address you obtained in the preceding step):
   - To view reports and dashboards, visit `http://<VM IP>:8080/MicroStrategy/servlet/mstrWeb`, and log in as an administrator.
   - To manage users and security roles, visit `http://<VM IP>:8080/MicroStrategy/servlet/mstrServerAdmin`, and log in as an administrator.

     Where <VM IP> is the hostname or ip-address where docker-compose is running.
   - To manage MSTR Web server settings, visit `http://<VM IP>:8080/MicroStrategy/servlet/mstrWebAdmin`, and log in as `admin`.

Procedure: 8. Installing MicroStrategy tools

Purpose: If you are setting up a development environment, you can use the instructions in this section to add administrative tools. This is required only if you expect to use these tools.

Prerequisites

Ensure that you have a copy of the latest MicroStrategy tools (for example, **MicroStrategy_11.1_Windows.zip**).

Steps

1. Extract the .zip archive to a temporary location on your hard drive, and run the **MICROSTRATEGY.EXE** file, to start the MicroStrategy Installation WIzard.
2. Follow the steps in the installation wizard.
   On the **Select Components** tab, clear all components *except* the following:
   - **MicroStrategy Developer Products**
   - **MicroStrategy Object Manager**
   - **MicroStrategy Command Manager**
   - **MicroStrategy Integrity Manager**
   - **MicroStrategy System Manager**
   - **MicroStrategy Analytics Module**.

   > **Important**
   > Install only the components indicated here. If you install additional components, you may encounter installation difficulties or performance issues.

Managing your environment

Use the procedures in this section to manage the environment.

Procedure: Stopping Genesys CX Insights containers and resetting the environment

Purpose: If, for any reason, you need to stop the containers, use the instructions in this section to stop the containers and clear any customizations.

Steps

1. Open a command terminal, such as PowerShell.
2. Change the current directory to the folder where you saved the Genesys CX Insights installation files. For example:

   ```
   cd C:\GCXI_temp
   ```

3. To stop the containers, execute the following command:

   ```
   docker-compose -f docker-compose.yml down
   ```

   Note that when you stop the containers, your customizations are preserved, and will still be available when you restart the containers.

4. To clear any customizations, and restore everything to the original state, first stop the containers, and then execute the following command:

   ```
   docker volume rm gcxi_mstr_log_01 gcxi_mstr_shared gcxi_gcxi_postgres
   ```

5. To restart the containers, execute the following command:

   ```
   docker-compose -f docker-compose.yml up
   ```

After installation

See Accessing CX Insights GUIs for information about accessing the reports using MicroStrategy Web. Note that, in the demo / sample database, data is available for a limited time period:

- For Genesys CX Insights reports, September 2015 to October 2016.
- For Genesys CX Insights for iWD reports, February 12, 2019 to February 21, 2019.

When you run reports in such environments, choose dates within that range, or simply remove the default value from the first prompt (Pre-set Date/Day) before you run the report.

Installing Genesys CX Insights - Podman Compose

GCXI is agnostic to container environments. It can work on top of Docker, Podman, or Kubernetes with any valid container runtime (like containerd or cri-o). The Podman setup is similar to a GCXI installation with Docker with minor syntactic differences between both types.

Prerequisites

GCXI deployment on Podman shares the same requirements to the host system, as GCXI on Docker.

- requirements to IPC / shared memory `sysctl` settings, namely `kernel.sem` and `vm.max_map_count`
- `selinux` must be set to permissive mode or disabled

Install Podman

1. Install Podman version 4.5 or greater. See Installation for more information.
2. Verify if the installation was successful by running `podman version`.
3. Enable Podman socket and verify.

```
systemctl enable --now podman.socket
systemctl status podman.socket
```

Install Podman Plugins

1. Install Podman plugins package. For RedHat 8/9, or CentOS Stream 8/9, the command will be: `yum install -y podman-plugins.x86_64`. See the Podman documentation for more information on installing plugins for your operating system.

Install Docker Compose

1. Install the docker-compose standalone binary following the instructions in Install Compose standalone.
2. Verify if the installation was successful by running `podman compose version`.

Prepare File Volumes

Similar to GCXI docker-compose deployment, GCXI on Podman uses certain folders on the host machine to store persistent data.

Before running GCXI on Podman, review all *host-path file volumes*, mentioned in the compose file, and ensure that the corresponding folders on the host

1. exist and
2. are owned by the user under which GCXI container runs (by default, user 500).

```
# compose file syntax reference
# https://docs.docker.com/compose/compose-file/

# excerpt from GCXI compose yaml file, describing host-path volumes
# note that particular host paths (left part before colon) are merely examples
# if needed, you may change them to whatever paths are convenient to you
volumes:
- "/genesys/gcxi_config:/genesys/gcxi_config"
- "/genesys/gcxi_shared:/genesys/gcxi_shared"
- "mstr_log_01:/mnt/log"

# NB: in this example, the log volume is defined as a temporary volume
# it will disappear after the GCXI deployment is brought down
# if you want to preserve GCXI logs permanently in the file system,
# you can redefine the log volume as a host-path volume
# '/mnt/log/gcxi' is an example path, you can choose any
volumes:
- "/mnt/log/gcxi:/mnt/log"

# make sure, that all host-path volumes folders, mentioned in the compose file:
# - exist
# - are owned by the container user (user 500 by default)

# for the example above, the needed commands will look like this:
mkdir -p "/genesys/gcxi_config" "/genesys/gcxi_shared"
chown -R 500:0 "/genesys/gcxi_config" "/genesys/gcxi_shared"
```

Prepare GCXI Images

Newer versions of Podman client work similar to Docker. Podman supports all necessary commands for working with images, like `podman pull` , `podman load`, or `podman tag`.

In most cases, GCXI images are shipped as tar.gz archives. Use `podman load` to create a usable OCI image, which may be consumed by Podman.

```
podman load -i gcxi.tar.gz
podman load -i gcxi_control.tar.gz

# (optional)
# after unpack, the images appear under full names, containing repo and version
# like 'pureengage-docker-production.jfrog.io/gcxi/gcxi:100.0.035.0000'
# if needed, retag them to more convenient names
# it allows to refer to these shorter names in your compose files or other tools
podman tag pureengage-docker-production.jfrog.io/gcxi/gcxi:100.0.035.0000 gcxi

# list images to ensure successful image load
podman images
```

Review GCXI Compose File

GCXI IP ships a sample **docker-compose.yaml** file.

This file may be **fully reused** by Podman versions > 4.5, no changes are needed. Configure GCXI variables in the compose file referring to the standard GCXI documentation. Ensure that the `image` references in your compose file are configured similar to how GCXI images are tagged on the host. General GCXI requirements about host IPC settings apply as usual.

Run GCXI on Podman Compose

1. Run GCXI on Podman Compose. `podman compose --file docker-compose.yml up`

Podman Compose almost fully mirrors behavior of Docker Compose, hence operating `podman compose` client shouldn’t comprise any difficulty.

Installing without Docker

Genesys recommends that you always deploy MicroStrategy and Genesys CX Insights using Docker and Kubernetes or OpenShift as described in this document. Genesys does not support installing Genesys CX Insights using manual methods; however, it is technically possible to do so. Genesys does not document the procedures necessary for such a deployment, which requires you to manually install MicroStrategy software, perform a database dump, and then deploy Genesys CX Insights. If you encounter difficulties in deploying MicroStrategy in such a non-Docker deployment, contact MicroStrategy for support.

Installing report editing software

If you plan to edit or create reports, you can install selected MicroStrategy software (MicroStrategy Developer or MicroStrategy Workstation) on a Windows machine, and use it as client software to connect to your Genesys Customer Experience Insights (Genesys CX Insights) installation. This page provides information about the environment required for such an installation, and the steps required to install these tools.

Microstrategy online documentation contains extensive information about other Windows software that you can optionally install, including information about system requirements, see the MicroStrategy ReadMe.

Before you begin

To install MicroStrategy Developer or MicroStrategy Workstation, the following are required:

- A compatible version of the MicroStrategy software. See the MicroStrategy downloads site, and refer to the Product Alert.
- A license key for MicroStrategy Developer. Contact your Genesys representative for assistance. MicroStrategy Workstation doesn't require a license key.
- A properly-prepared system with a supported release of Windows software. See the table **Minimum requirements for selected MicroStrategy Windows components**.

Minimum requirements for selected MicroStrategy Windows components

Component	Processor	RAM	Disk Space
MicroStrategy Desktop on Windows	x86 or x64 compatible	4 GB (minimum)	8 GB (minimum)
MicroStrategy Developer Products	x86 or x64 compatible	2 GB (minimum)	0.25 GB
MicroStrategy Object Manager	x86 or x64 compatible	2 GB (minimum)	0.25 GB
MicroStrategy Command Manager	x86 or x64 compatible	2 GB (minimum)	0.25 GB
MicroStrategy Integrity Manager	x64 compatible	2 GB (minimum)	0.25 GB
These values apply at the time of writing. For the latest information, or for information about requirements for other components, see MicroStrategy System Requirements and the MicroStrategy ReadMe for the MicroStrategy release you plan to install.			

You can install either MicroStrategy Developer, or MicroStrategy Workstation, or both.

Installing MicroStrategy Developer

To use MicroStrategy Developer, you must install the following additional MicroStrategy components:

- MicroStrategy Developer Products
- MicroStrategy Object Manager
- MicroStrategy Command Manager
- MicroStrategy Integrity Manager

Note that you should not install components other than those listed here, as the hardware and system requirements described on this page will not support a full install. For information and instructions, see:

- About MicroStrategy Developer — Microstrategy Developer ReadMe
- Installing MicroStrategy Developer — Installing and Upgrading MicroStrategy.

After completing the steps on this page, complete the following post-Installation steps:

- After Installing Genesys CX Insights

Installing MicroStrategy Workstation

MicroStrategy Workstation is available as a separate installation package, provided by Genesys, and installation instructions are provided by MicroStrategy. For information about downloading the package, see Before you begin: Installation packages.
For more information about MicroStrategy Workstation, including installation instructions, see: MicroStrategy Workstation Help.

After Installing Genesys CX Insights

After you have installed Genesys Customer Experience Insights (Genesys CX Insights), you must manually perform additional setup steps before you operate the Genesys CX Insights reports.

Getting Started

Check to ensure that you can now view reports and dashboards. You can also optionally hide unwanted objects, and check the GCXI release and schema number.

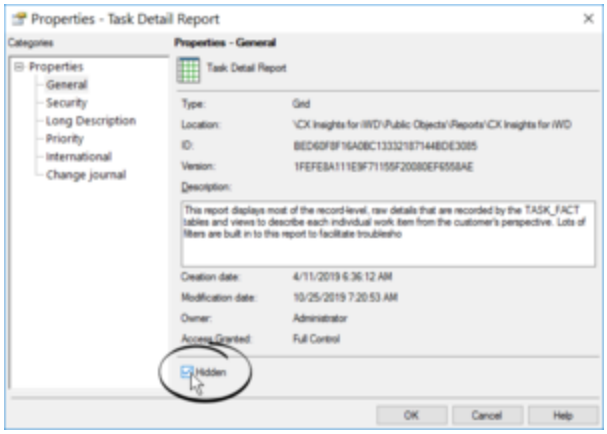
View the reports

Viewing Historical Reports

To view (or edit) the Genesys CX Insights historical reports, open MicroStrategy Web by pointing your web browser to `http://<servername>:<port>/MicroStrategy/servlet/mstrWeb*`, where `<servername>:<port>` are the server name and port provided by your administrator.

Verify that an assortment of report folders are present (Agents, Business Results, Callback, and so on) and that each one contains one or more reports. Note that, before you can view a report, you must run the report in order to populate data. For more information, see Accessing CX Insights GUIs and the *Genesys CX Insights User's Guide*.

*Redirecting the base URL — By default, users can enter simply `http://<server>:8080/MicroStrategy` (where `<server>` is the URL of your server) instead of `http://<server>:8080/MicroStrategy/servlet/mstrWeb`. Optionally, you can further simplify this by creating a URL redirect that allows users to access reports by entering `http://<server>`. For more information, see the *Other Properties* section of Procedure: Enter database information in the properties file.



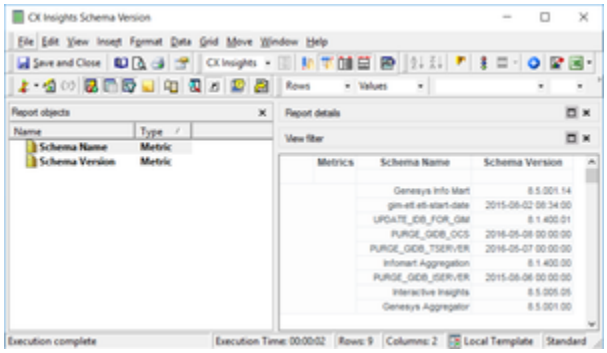
Hiding a Report

Hide unwanted reports

Some reports are needed only in certain scenarios, or there may be reports you don't want your users to see, for whatever reason. Optionally, you can hide unwanted reports (or other objects):

1. In MicroStrategy Developer, open your project, and navigate to the location where the report is stored.
2. Right-click the report name, and select **Properties**.
3. Click the **Hidden** check box.
4. Click **OK**.

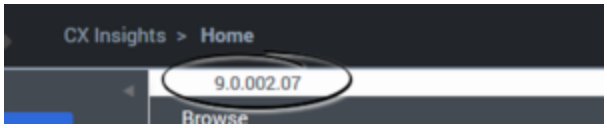
View the schema version



CX Insights Schema Version dialog

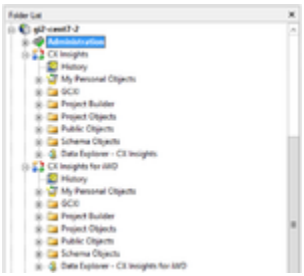
In MicroStrategy Developer, select **CX Insights** > **Public Objects** > **Reports**, and open the object **CX Insights Schema Version**. The *CX Insights Schema Version* dialog appears, where you can can view the current schema version for various Genesys components.

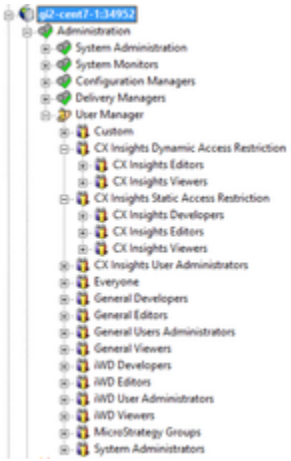
View the Genesys CX Insights Release number



CX Insights Release Number

You can view the Genesys CX Insights release number when you open the project in MicroStrategy Web. It appears just below the breadcrumbs, when you first open the project, as shown in the figure **CX Insights Release Number**. You can also view the release number when you select the project in MicroStrategy Developer.


GCXI folders



CX Insights User Manager

View the project

If you have installed the optional report editing software (see Installing report editing software), you can view and edit the CX Insights project in MicroStrategy Developer, a desktop application available on the system where you installed MicroStrategy. For information about MicroStrategy Developer, see the Microstrategy ReadMe, and other MicroStrategy documentation (see Additional Resources for links to many useful MicroStrategy documents).

> **Important**
>
> Early releases of Genesys CX Insights comprised one project, called **Genesys CX Insights**. Beginning with release 9.0.010, a second project, **CX Insights for iWD** is included for iWD customers. The **Genesys CX Insights for iWD** project has structure and features similar to the **CX Insights** project. For additional information about **CX Insights for iWD**, see CX Insights for iWD reports.

Note that Genesys does not support customization of the underlying metadata, but does support customization of the reports. You can create your own reports, though Genesys recommends that you first familiarize yourself with the existing reports, as it is easier to modify one of them, than it is to start from scratch. For more information about the included reports, and how to customize or create reports, see the *Genesys CX Insights User's Guide*.

Genesys CX Insights is organized in a folder hierarchy, as shown in the figure **GCXI folders**. The subfolders you will most often be concerned with include:

- **GCXI > Administration > User Manager** — Note that the **User Manager** folder is reorganized in release 9.0.010.
- **GCXI > CX Insights > GCXI**
- **GCXI > CX Insights for iWD > iWD** (in release 9.0.010 and later)

The following table (**Most-used folders**) list the elements you will most often access in the MicroStrategy Developer file list:

Most-used folders

Administration	This folder contains administration and management tools you can use to manage such elements as user objects, database instances, and connections.
Administration > User Manager — This folder contains groups. Each group can contain users, or other groups.	Often-used groups in **release 9.0.010 and later:** • The following User Groups in the **User Manager >CX Insights Dynamic Access Restrictions** folder: CX Insights Editors, CX Insights Viewers • The following User Groups in the **User Manager >CX Insights Static Access Restrictions'** folder: CX Insights Developers, CX Insights Editors, CX Insights Viewers • The following User Groups in the **User Manager** folder: General Developers, General Editors, General Viewers, General Users Administrators, iWD Developers, iWD Editors, iWD Users Administrators, iWD Viewers Most other groups are seldom-used in Genesys CX Insights deployments. For more information about users and groups, see Users and Groups and Managing Users, Groups, and Privileges in

	the *Genesys CX Insights User's Guide*.
	CX Insights project (Includes GCXI reports, and the objects used to build them)
CX Insights — The root of the CX Insights project.	This folder contains all GCXI report objects, as well as administration and management tools you can use to manage such elements as user objects, database instances, and connections.
CX Insights > GCXI	Various subfolders that contain the objects (metrics, attributes, and prompts) that make up each report.
CX Insights > Public Objects > Reports	More than fifty reports and dashboards in subfolders within the **CX Insights > Public Objects > Reports > CX Insights > Reports** folder. Subfolders include: **Agents**, **Business Results**, **Callback**, **Chat**, **Dashboards** , **Designer**, **Details**, **Outbound Contact**, **Queues**, **Callback**, and various other folders. Dashboards are found in many of the report folders, in addition to those in the **Dashboards** folder.
CX Insights for iWD (release 9.0.010 and later — includes iWD reports, and the objects used to build them)	
CX Insights for iWD — The root of the CX Insights for iWD project.	This folder contains all iWD report objects, as well as administration and management tools you can use to manage such elements as user objects, database instances, and connections.
CX Insights for iWD > iWD	Various subfolders that contain the objects (metrics, attributes, and prompts) that make up each report.
CX Insights for iWD > Public Objects > Reports	The iWD reports, within the subfolder **CX Insights for iWD > Public Objects > Reports > CX Insights for iWD**.

*For more information about data access restrictions, see Data Access Restrictions and subsequent sections on this page.

Readying Genesys Info Mart for Aggregation

A Genesys Info Mart 8.5 installation that has the Reporting and Analytics Aggregates (RAA) option deployed contains the tables and views that are referenced by Genesys CX Insights reports. To prepare the Genesys Info Mart environment for Genesys CX Insights operation, you must perform additional setup steps, including:

- **Set Aggregation-Related Configuration Options**:
  To enable aggregation, you must appropriately set aggregation-related configuration options (such as **aggregation-engine-class-name**, **run-aggregates**, and business-specific aggregation

thresholds) in the Genesys Info Mart application object in the Genesys Administration Extension (GAX) Configuration Manager. These options are described in the How Do I Configure Genesys Info Mart for Aggregation? section of the *Reporting and Analytics Aggregates Deployment Guide*. For information about how to configure options using GAX Configuration Manager, see the *Genesys Administrator Extension Help*.

> **Tip**
>
> There are two GUIs called *Configuration Manager* discussed on this page. One is part of GAX (so is referred to on this page as GAX Configuration Manager), and is used to configure options, such as **run-aggregates**. The other is part of MicroStrategy Developer, and is used to configure database information in MicroStrategy.

Utility Views Specific to Genesys CX Insights

Running aggregation for the first time executes an internal script against your Genesys Info Mart database to set up the necessary views that facilitate data processing for the Genesys CX Insights reports.

Genesys Info Mart Multi-Tenant Environments —For Genesys Info Mart environments that contain more than one tenant, run RAA with the **updateAliases** runtime parameter to create tenant views of Genesys CX Insights objects. For a description of this parameter and an example of its use, refer to the *Reporting and Analytics Aggregates Deployment Guide* and the *Reporting and Analytics Aggregates User's Guide*, respectively.

Setting Up Attached Data

Genesys CX Insights reports are based on the configuration of user data in your environment — user data that is highly customizable within any given environment. To use the Genesys CX Insights reports without modifying the CX Insights Project or metric definitions, you must configure user-data data structures within Genesys Info Mart in a specific manner. For more information, contact your Genesys representative.

Linking the CX Insights Project to Your Data Mart

The Genesys CX Insights reports call upon metrics that are predefined in the CX Insights Project, but they are not pre-connected to your specific Genesys data source out of the box. You must define such a connection and assign it so that the reports that reference these metrics can pull contact center

data from your Info Mart database.

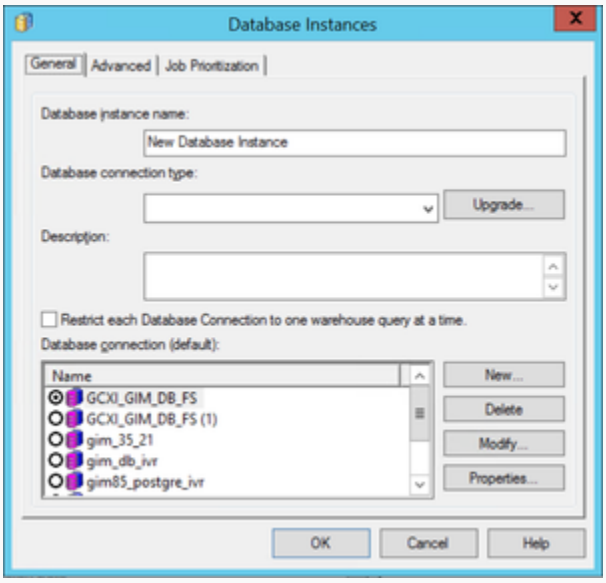
Use the following procedures to link the CX Insights Project to your Info Mart database.

Establishing communication between Genesys CX Insights / MicroStrategy and your database is an essential first step in configuring the software for reporting and analysis of your data. This section explains the steps required to set up this communication.

Procedure: Creating a new database instance using the default database connection

Purpose: Use this procedure to create a database instance, which is a MicroStrategy object that represents a connection to a data source. A database instance specifies connection information, including the name of the data source, login credentials, and other information about the data source.

Steps



Create a new database instance

1. Open MicroStrategy Developer.
2. In the **Folder List** list, expand **Administration** > **Configuration Managers**, and select **Database Instances**.
3. On the **File** menu, select **New** > **Database Instance**. The **Database Instances** editor appears.
4. In the **Database Instances** list, select the **GCXI_GIM_DB connection**. Instead of using the

default connection, you can optionally create a new connection by following the steps in Creating a new database connection.

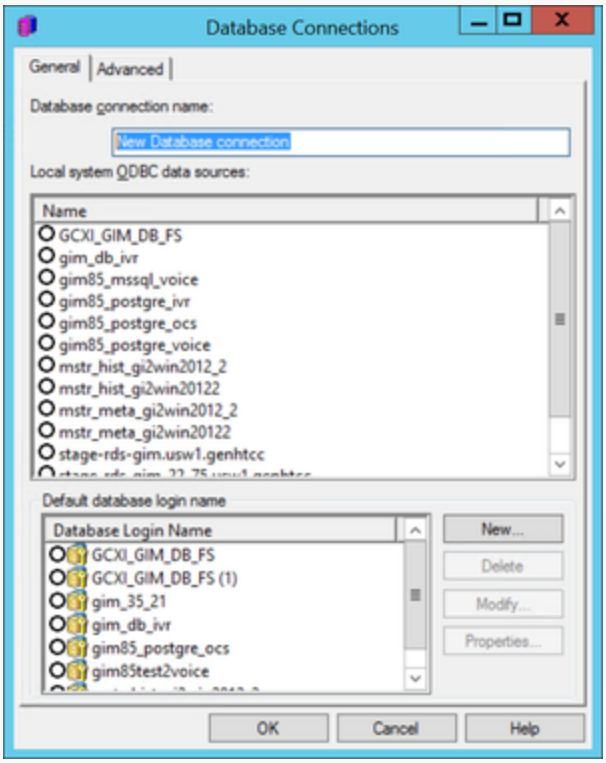
5. On the **General** tab, in the **Database instance name** field, type a name for the database instance, and in the Database connection (default) list, select the default data source connection.
6. From the **Database connection type** list, select a data source connection type suitable for the data source hosting your database.
7. On the **Advanced** tab, optionally configure additional options for the database instance.
8. On the **Job Prioritization tab**, optionally configure how jobs are prioritized for the database instance. For more information about prioritization, see the MicroStrategy System Administration Help.
9. Click **OK** to save your changes and close the dialog.

> **Tip**
>
> To learn more about the specific options you can configure in the **Database Instances** editor, click **Help**.

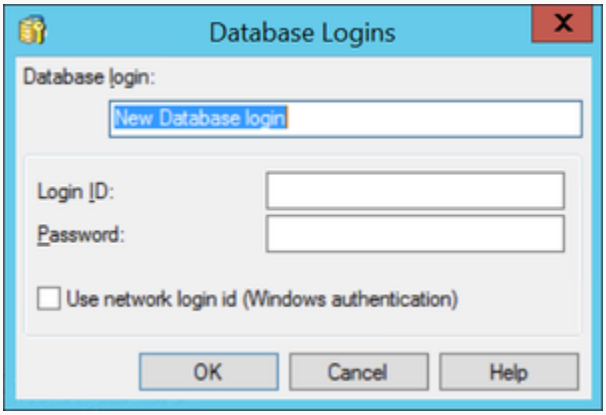
Procedure: Creating a new database connection

Purpose: Use this procedure to define a new connection, which you can use to link the CX Insights Project objects to the tables in your Data Mart. The database connection specifies the Date Source Name (DSN) and database login information used to access the data source. Alternatively, you can reuse the default connection by following the steps in Creating a new database instance using the default database connection.

Steps



Database Connections



Database Logins

1. Open MicroStrategy Developer.
2. In the **Folder List** list, expand **Administration** > **Configuration Managers**, and select **Database Instances**.

3. On the **File** menu, select **New** > **Database Instance**. The **Database Instances** editor appears.
4. Next to the **Database connection (default)** list, click **New**. The **Database Connections** editor appears.
5. On the **General** tab, in the **Database connection name** field, type a name to identify the database connection.
6. From the **Local system ODBC data sources** list, select the data source name.
7. On the **Advanced** tab, optionally configure additional options as required for the database to which you are connecting.
8. On the **General** tab, in the **Default database login name** list, select the default database login and click **OK**.

If the database login you require does not exist, you can create one as follows:

1. Next to **Default database login name**, click **New**.
2. In the **Database login** field, type a name for the database login.
3. Choose one of the following:
   - In the **Login ID** field, type the user name for the database login, and in the **Password** field, type the password associated with the user name.
   - Select **Use network login ID** to connect to the data source using the network user credentials that are used to run Intelligence Server.
4. Click **OK**.

> **Tip**
> To learn more about the specific options that you can configure (in the **Database Connections** editor, for example), click **Help**.

Users and Groups

The Genesys CX Insights installation routine silently deploys a variety of Genesys CX Insights objects, including Genesys CX Insights groups and users.

> **Important**
>
> Beginning with release 9.0.013, the default user accounts (Developer, Editor, Viewer), are disabled by default. A new container management variable, `GCXI_USERS_ENABLED=false|true`, is added, which you can use to enable the default accounts.

Some groups are distinguished by type, as shown in the table **Group types**.

Table: Group types

Group type	Capabilities
General (For example, **General Developers**, **General Users Administrators**)	Members of these groups have access to all installed projects (GCXI, iWD, and any others).
CX Insights (For example, **CX Insight Developers**, **CX Insights User Administrators**)	Members of these groups have access only to the CX Insights project (other projects, for example iWD, is not visible to them). Members of **CX Insights User Administrators** are able to manage only CX Insights groups (and users in these groups).
iWD (For example, **iWD Developers**, **iWD User Administrators**)	Members of these groups have access only to the CX Insights for iWD project (other projects, for example iWD, is not visible to them). Members of **iWD Administrators** are able to manage only CX Insights for iWD groups (and users in these groups).

You can import these objects with their permissions applied to project elements, or create the objects yourself from scratch and assign permissions to various objects by following the instructions in the following procedures. Each object in the Genesys CX Insights project has an Access Control List (ACL), which dictates which users can view or modify the object. The Table, **Mapping of Access Levels to selected objects**, lists the user security properties of objects that the Genesys CX Insights installation routine sets.

Table: Mapping of Access Levels to selected objects

Object type	User Group	Object Permission	Permission passed to children
Folder objects within the project	Administrator	Full Control	Default
	Everyone	Custom	View
	Public / Guest	Custom	View
Default report objects	CX Insights report developers	Custom	
	CX Insights report editors	Custom	
	CX Insights report viewers	Custom	
	System Administrators	Modify	
Default objects in the	CX Insights report	Custom	

Public Objects folder, including metrics, and prompts	developers		
	Everyone	View	
	Public / Guest	View	
	System Administrators	Modify	
Default objects in the Schema Objects folder, including attributes	Administrator	Full Control	
	CX Insights access restrictions	Custom	
	CX Insights report developers	Custom	
	Everyone	View	
	MicroStrategy Architect	View	
	Public / Guest	View	
	System Administrators	Modify	

Controlling access to Genesys CX Insights

Genesys CX Insights, through MicroStrategy, provides several tools to help you control access the historical reports. The easiest way to assign privileges to users is by assigning the user to a group that has the desired privilege; the user inherits permissions from the group object. You can also control access at the object level using an Access Control List (ACL).

> **Important**
>
> In scenarios where agents or queues are members of more than one group, and access restrictions are configured for all groups of which the agent or queue is the member, data can be double-counted in reports.

By default, a newly created object has an ACL consisting of:

- The user who created the object: Full Control
- Permissions for other users: as inherited from the parent folder

Procedure: Setting Access at the object level

Purpose: Use this procedure to manually set permissions on the objects used by Genesys CX Insights, including the Project, the **CX Insights** folder and connection, and even the

MicroStrategy applications. You must be an administrative user to make these changes.

Steps

1. In MicroStrategy Developer, log in to a project source (as a user with the 'Create And Edit Users And Groups' privilege).
2. Expand **Administration** > **User Manager**.
3. Right-click an object, and select **Properties**.
4. In the **Categories** list, click **Security**. Modify access as required, and click **OK**.

For detailed information about what access level settings you can apply, see the MicroStrategy documentation.

Procedure: Creating GCXI Users

Purpose: Use this procedure to create users and assign them to groups.

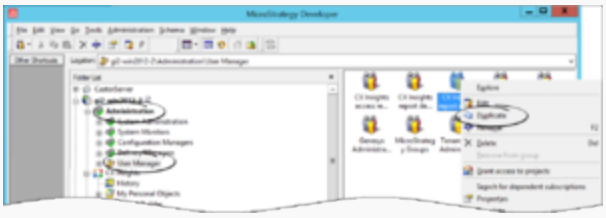
Steps

1. In MicroStrategy Developer, log in to a project source (as a user with the 'Create And Edit Users And Groups' privilege).
2. Expand **Administration** > **User Manager**, and then expand a group in which to add the new user. If you do not want the user to be a member of a group, select **Everyone**.
3. Select the menu item **File** > **New** > **User**. The **User Editor** opens.
4. Specify a user name and other information as appropriate, and click **OK**.

Procedure: Creating GCXI Groups

Purpose: In many cases, you can simply assign your users to the default groups. Alternatively, use this procedure to make a copy of an existing group, and modify the copy.

Steps



Duplicating a user group

1. In Microstrategy Developer, log in to a project source (as a user with the 'Create And Edit Users And Groups' privilege).
2. Expand **Administration** > **User Manager**, and then right-click an existing group, and choose **duplicate**. The **Group Editor** appears.
3. Right-click the new group, and click **Edit**. Edit the name of the new group, and other information as appropriate.
4. Click **OK**.

About Data-Access Restrictions for Multi-Tenant Environments

In addition to the permissions that you can set to control access to various MicroStrategy repository elements, you can also restrict the data that users can access by limiting the objects, rows, query types, and connections that are available to users. Through the use of these restrictions, you can control what data users see in the CX Insights reports.

Use this feature if your data source stores data for more than one tenant. For instance, within one project, you can define several connections—each of which accesses a different tenant view within the same Info Mart—and then create and apply connection restrictions to each tenant to ensure that its users see only the data that is pertinent to that tenant.

The credentials that a user enters when logging in to MicroStrategy identify the user (and hence the user group) and the access permissions that are assigned to that user within the repository; the restriction defines which connection the user can use to access data within a specific CX Insights Project.

The benefits of this one-project approach include:

- Consistency in metric definitions across the enterprise.
- Reduced maintenance costs—having to manage only one project (instead of one per tenant).
- Single source.
- Optimized use of network resources.

Genesys Info Mart supports several methods of configuring multi-tenant environments, including:

- A separate schema for each tenant.
- A separate schema for each group of tenants.
- One database/one schema for all tenants (where each tenant can see other tenants' data).

Configuration depends largely on the capabilities that are provided by your chosen RDBMS and on the data access security measures that are established within your enterprise. Please refer to the *Genesys Info Mart Deployment Guide* for further information.

About Integrated Data Access Restrictions

Data access restrictions are integrated with data access roles. These restrictions control access to objects within the Info Mart database so that MicroStrategy users who are members of MicroStrategy groups with associated access restrictions see data only for appropriate contact center resource groups (Agent Groups or Queue Groups that are configured in the Configuration Layer). Beginning with release 9.0.010, there are two types of these restrictions:

- **Static Access Restrictions** — Enable you to configure a list of objects for which no data appears when reports are viewed by users who are members of restricted groups. For example, you can use this feature to prevent group members from viewing data for 'system' objects (such as Queue/Queue Groups).
- **Dynamic Access Restrictions** — Enable you to restrict access to data based on each BI user name and the attributes you configure to describe the user's geographical location, line of business, or organizational role. For example, you can use this feature to ensure that a supervisor sees data only from agents in specified locations, on specified teams.

Limitations

The following limitations apply to data access restrictions:

- Agent hierarchy — Normally, dynamic access restrictions are applied everywhere that objects from the **Agent/Queue/Other** and **Agent/Queue Groups** are used. However, in the Interaction Flow Report (in release 9.0.009.00 and later) restrictions are applied only on Target resources; Source data from the **Agent/Queue/Other** and **Agent/Queue Groups** hierarchies is displayed in the report without access restrictions.
- Co-browse hierarchy — In the Co-Browse Details report, the Agent Name prompt and attribute always show all agents, regardless of Dynamic Access Restriction filters.
- Details hierarchy — The following reports, in the **Details** folder, have no access restrictions on specific objects, as follows:
  - Interaction Handling Attempt Report — Access restrictions are applied only on Agents. Queue data is displayed without restrictions, including the lists of values in prompts, and queues in the report itself.
  - Transfer Details Report — In release 9.0.008.00 and later, restrictions are applied only on Source Agents; Target Agent and Queue data appears without access restrictions, including the lists of

values in prompts.

  - Agent Group Membership Details Report — is not under any restriction, and always shows data for all groups and agents.

- **Reports with Agent Group prompts** — In some scenarios where agents are members in more than one group, data that users expect to see does not appear in a report. This can happen when the user who runs the report is permitted to see the agent's data, but is not permitted to see data for a group of which the agent is a member. In this scenario, when the user runs the report, the restricted group appears on the prompts page, allowing the user to select the group; however data for that group does not appear in the report.
- **Drilling to groups**— In some scenarios, after a user runs a report and then drills from agent to agent group, the report shows duplicated data rows for agents who are members of more than one group. This also causes totals in the report to be incorrect.
- **Groups reports** — In scenarios where agents or queues are members of more than one group, and at least one of the groups has Data Access Visibility (DAV) attributes configured, Group reports can duplicate data for those agents/queues, attributing it to each of the groups in which the agent/queue is a member. This also causes totals in the report to be incorrect.

Configuring Data Access Restrictions

You can customize Dynamic Access Restrictions by configuring the following Data Access Visibility (DAV) attributes, which are available on each object's Annex tab:

- ORG (Organizational Role)
- GEO (Geographic Location)
- LOB (Line of Business)

You restrict access to data by defining values on the Annex tab, as follows:

- For each Person: BI login, plus one or more DAV attributes
- For each contact center group: one or more DAV attributes

As long as a user has at least one DAV attribute that matches a group, then that user can see data from that group. For example,

- If the following values are configured:
  - Agent Group1 has the following annex value: RPT_GEO=Daly City
  - Agent Group2 has the following annex value: RPT_GEO=San Francisco
  - Agent Supervisor1 has the following annex value: RPT_GEO=Daly City
  - Agent Supervisor2 has the following annex value: RPT_GEO=San Francisco
- Then, when Agent Supervisor1 runs a report, the report contains data from Agent Group1, but not data from Agent Group2. The reverse is true for Agent_Supervisor2.

Data access restrictions use a small amount of system resources, so configuring them can result in a

slight decrease in system performance.

Procedure: Configuring Access Restrictions

Purpose: Define DAV attributes using GAX Configuration Manager (for Dynamic Access Restrictions only), and define access restrictions using MicroStrategy Developer.

- For information about working in GAX Configuration Manager, see Configuration Manager.
- For information about MicroStrategy Developer, and other MicroStrategy tools, see the resources listed on the Additional Resources page.

> ### Tip
> There are two GUIs called *Configuration Manager* discussed on this page. One is part of GAX (so is referred to on this page as GAX Configuration Manager), and is used to configure options, such as **run-aggregates**. The other is part of MicroStrategy Developer, and is used to configure database information in MicroStrategy.

Steps

1. To ensure that access restrictions are enabled, use either of the following methods to manage security filters:
   - In any Genesys CX Insights release, using MicroStrategy Developer:
     1. Open the User Manager.
     2. Right-click the **CX Insights Dynamic Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
     3. Click **Security Filters**, then click **View**.
     4. Ensure that the Dynamic Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.
     5. Right-click the **CX Insights Static Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
     6. Click **Security Filters**, then click **View**.
     7. Ensure that the Static Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.

        OR
   - In Genesys CX Insights release **9.0.010 or later**, using the web interface:

1. Open the following URL in your web browser:

   ```
   http://<Server>:<Port>/MicroStrategy/servlet/mstrServerAdmin
   ```

   where <Server>:<Port> is the IP address and port of your Genesys CX Insights deployment.
2. Click the **Server** icon, and open the **User Manager**.
3. Right-click the **CX Insights Dynamic Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
4. Select the **Security Filters** tab.
5. Ensure that the Dynamic Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.
6. Right-click the **CX Insights Static Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
7. Select the **Security Filters** tab.
8. Ensure that the Static Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.

2. For both Dynamic Access Restrictions and Static Access Restrictions — to assign users to the appropriate groups, complete the following actions using MicroStrategy Developer:
   1. Create users as required.
   2. Assign each user to the following groups:
      - For Dynamic Access Restrictions, choose from: **CX Insights Dynamic Access Restriction** and **CX Insights Developers**, or **CX Insights Editors**, or **CX Insights Viewers**.
      - For Static Access Restrictions, choose from: **CX Insights Static Access Restriction** and **CX Insights Developers**, or **CX Insights Editors**, or **CX Insights Viewers**.
3. For Dynamic Access Restrictions — define DAV attributes in GAX Configuration Manager, as follows:
   1. Open **View > Options**, and ensure that **Show Annex tab in object properties** is selected. Perform the following steps for each user (Person):
      1. If it is not already present, add the RPT section.
      2. Within the RPT section, add an option with:
         - **Option Name = BOE_USER**
         - **Option Value = <username>**
      3. If they are not already present, add one or more of the following sections:
         - **RPT_GEO**
         - **RPT_ORG**

- **RPT_LOB**

4. Within each of the sections you added, assign suitable options. For example, within the RPT_GEO section, you might add an option and assign it an Option Name that describes the geographical location of a group, such as `Daly City`.

   Neither Genesys Info Mart nor Genesys CX Insights processes the Option Value for options in the [RPT_GEO], [RPT_ORG], or [RPT_LOB] sections, so you can leave the option value blank, and enter only the option name (unless the Configuration Server installed in your environment requires a value, as is the case in Configuration Server 7.6 and earlier).

2. Perform the following steps for each contact center Group (Agent Groups and DN [ACD Queue] Groups):

   1. If they are not already present, add one or more of the following sections:

      - **RPT_GEO**
      - **RPT_ORG**
      - **RPT_LOB**

   2. Within each of the sections you added, assign suitable options. For example, within the **RPT_GEO** section, add an option and give it an **Option Name** that describes the geographical location of a group, such as `Daly City`.

Link to video

Dynamic Access Restriction Configuration Example

The following example creates restrictions so that when the user **cxiuser1** views Genesys CX Insights reports, the data in the reports comes only from **Agent Group 1** (and agents in that group) and **Queue Group 1** (and queues in that group).

1. To ensure that access restrictions are enabled, use either of the following methods to manage security filters:

   - In any Genesys CX Insights release, using MicroStrategy Developer:

     1. Open the User Manager.
     2. Right-click the **CX Insights Dynamic Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
     3. Click **Security Filters**, then click **View**.
     4. Ensure that the Dynamic Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.
     5. Right-click the **CX Insights Static Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
     6. Click **Security Filters**, then click **View**.

7. Ensure that the Static Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.

OR

- In Genesys CX Insights release **9.0.010 or later**, using the web interface:
  1. Open the following URL in your web browser:

     ```
     http://<Server>:<Port>/MicroStrategy/servlet/mstrServerAdmin
     ```

     where <Server>:<Port> is the IP address and port of your Genesys CX Insights deployment.
  2. Click the **Server** icon, and open the **User Manager**.
  3. Right-click the **CX Insights Dynamic Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
  4. Select the **Security Filters** tab.
  5. Ensure that the Dynamic Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.
  6. Right-click the **CX Insights Static Access Restriction** user group, and select **Edit**. The **Group Editor** appears.
  7. Select the **Security Filters** tab.
  8. Ensure that the Static Access Restriction type is in the **Selected** list (the right-hand list). If it is not, add it from the **Available** list, and click **OK**, then **OK** again.

2. Create the user **cxiuser1**, and add the newly created user to the following groups: **CX Insights Dynamic Access Restriction** and **CX Insights Developers**, or **CX Insights Editors**, or **CX Insights Viewers**.
3. Log in to GAX Configuration Manager, and in the Annex of **cmperson1**, create the section **RPT** with option **BOE_USER=cxiuser1** and section **RPT_GEO** with option **Daly City=<any value>** as follows:

   ```
   [RPT]
   BOE_USER=cxiuser1
   [RPT_GEO]
   Daly City=<any value>
   ```

4. In the Annex of **Agent Group 1**, create the section **RPT_GEO**, and add the option **Daly City=<any value>**, as follows:

   ```
   [RPT_GEO]
   Daly City=<any value>
   ```

5. In the Annex of **Queue Group 1**, create the section **RPT_GEO**, and add the option **Daly City=<any value>**, as follows:

   ```
   [RPT_GEO]
   Daly City=<any value>
   ```

6. Run Genesys Info Mart and execute one ETL cycle. All data for objects with configured Annex are added in GIM tables: RESOURCE_ANNEX and GROUP_ANNEX.

> **Tip**
>
> Genesys CX Insights relies on Interaction Concentrator and Genesys Info Mart to populate the RESOURCE_ANNEX and GROUP_ANNEX tables. Refer to the *Interaction Concentrator Deployment Guide* and *Genesys Info Mart Deployment Guide* for information about how to configure the population of Annex data (using the Interaction Concentrator **cfg-annex** option).

The user **cxiuser1** now sees report data only from Agent Group 1 and Queue Group 1.

User and account management

Genesys recommends that you immediately change the default Microstrategy administrator password. For information about how to do so, and other procedures needed to create and manage users, see Managing the MicroStrategy environment.

Accessing CX Insights GUIs

This page provides information about how to access each of the relevant Genesys Customer Experience Insights (Genesys CX Insights) application (GUI)s. If you require more information than is given on this page, see the *Genesys CX Insights User's Guide* and MicroStrategy Product Documentation.

- Access Genesys CX Insights reports using the MicroStrategy Web interface.
- Access the CX Insights Project (advanced users) through the MicroStrategy Developer interface.

MicroStrategy Web

MicroStrategy Web is the user interface most often used for accessing, managing, and running the Genesys CX Insights reports. MicroStrategy Web certifies the latest versions, at the time of release, for the following web browsers:

- Apple Safari
- Google Chrome (Windows and iOS)
- Microsoft Edge
- Microsoft Internet Explorer (Versions 9 and 10 are supported, but are not certified)
- Mozilla Firefox

To view updated information about supported browsers, see the MicroStrategy ReadMe. To start this application, you must have the name of the Web server that has been established by your administrator, or the complete URL if your administrator configured other than the default parameters and path, and you must have valid user credentials. The default path is: `http://<server>:8080/MicroStrategy/servlet/mstrWeb` where <server> is the name of the server provided by your administrator.

Video: How do I generate Historical Reports using Genesys CX Insights?

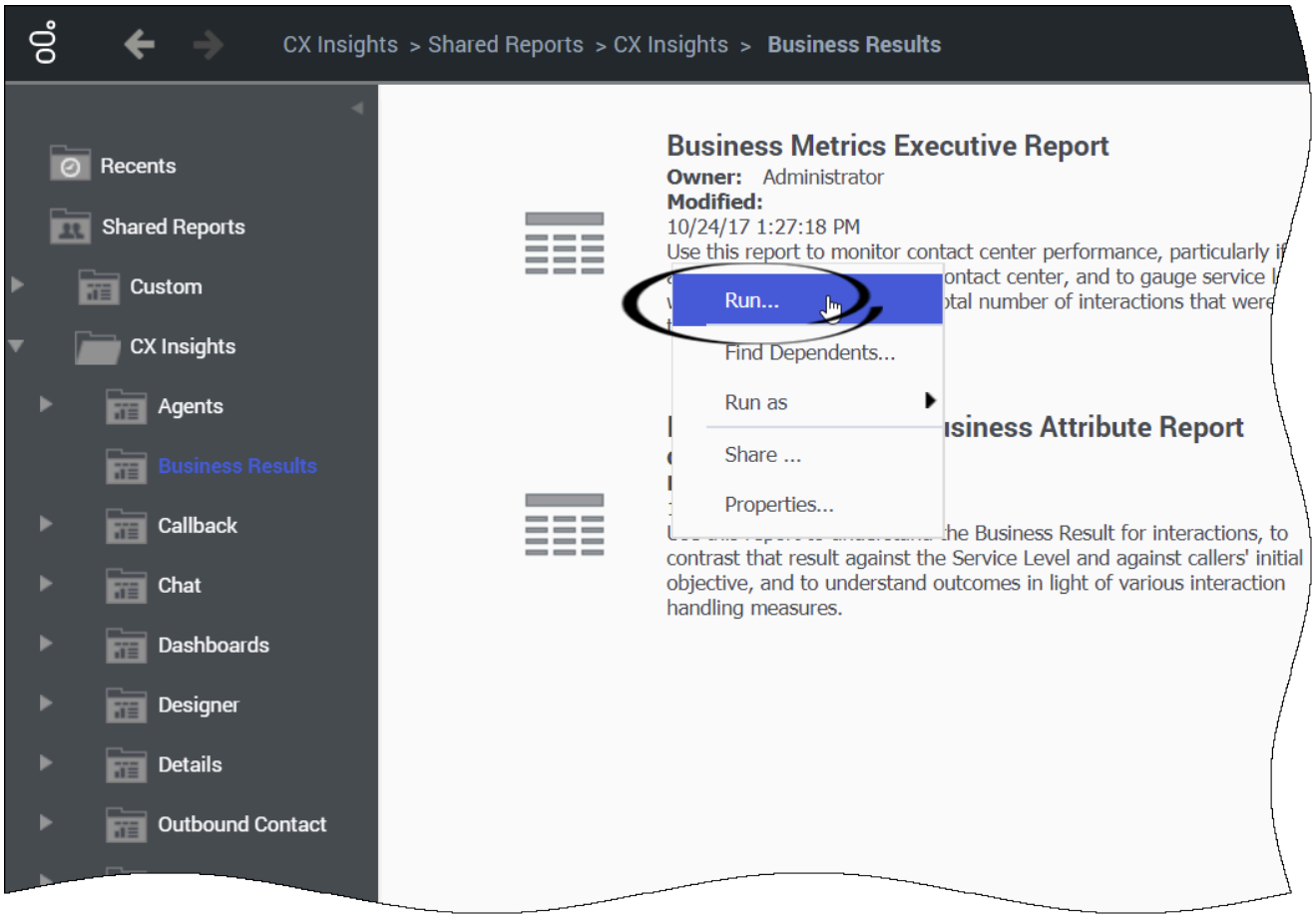
Link to video

This video describes how to generate historical reports using Genesys CX Insights. Open MicroStrategy Web in your web browser (`http://<server>:8080/MicroStrategy/servlet/mstrWeb`), and then click the video for more information.

[+] Tip: What is a Historical Report?

Historical Reports are reports that track contact center and agent performance over a period of time. How far back in time you can look varies depending on the size and complexity of your contact center. By contrast, **Real-time Reporting** provides information about interactions that are taking place *right now* in the contact center.

Using MicroStrategy Web to generate and view reports



1. In your web browser, open MicroStrategy Web by entering the URL as follows: `http://<server>:8080/MicroStrategy/servlet/mstrWeb`, where `<server>` is the URL of your server.
2. When prompted, enter your user name and password.
3. The Genesys CX Insights page appears. Click **Shared Reports** > **CX Insights**.
4. Reports are divided into subfolders based on function; select a sub-folder, for example **Business Results**.

5. From the listed reports, either double-click a report, or right-click and choose **Run**. For example, **Business Metrics Executive Report**. The prompts for that report appear.
6. Select a date or date range, and optionally make selections for other prompts.
   Note that, if you have installed a demo/development environment, data is available for a limited period:
   - For Genesys CX Insights reports, September 2015 to October 2016.
   - For Genesys CX Insights for iWD reports, February 12, 2019 to February 21, 2019.

   When you run reports in such environments, choose dates within that range, or simply remove the default value from the first prompt (Pre-set Date/Day) before you run the report.
7. Click **Run Report**. The report appears.

Note that you can filter, drill, and otherwise interact with many report values.

Many reports offer a long list of prompts, but you don't have to make selections at all of those prompts.

For most reports, you can simply select a date or date range, and click **Run Report** to generate the report. Before you do, note that the default **Start Date** and **End Date** encompass the entire current year; depending on your environment, this may not be a suitable range. For more information:

- For descriptions and samples of individual reports, including descriptions of the prompts, metrics, and attributes used in the Genesys CX Insights reports, and more information about using MicroStrategy Web, see the *Genesys CX Insights User's Guide*.
- For additional documentation about MicroStrategy Web , click Help (?)

The Contact Center Dashboard on a mobile device

Mobile support

You can also access Genesys CX Insights dashboards from your Apple iPad. Genesys recommends you use only the dashboards in this fashion, and does not recommend using reports from a mobile device, as they are not designed for small screens.

To enable mobile support, you must:

- ask your administrator to verify that the MicroStrategy server is configured to support mobile access,

- download the app on your Apple iPad,
- configure the app with access information and credentials for your Genesys CX Insights deployment.

For more information, see the MicroStrategy website.

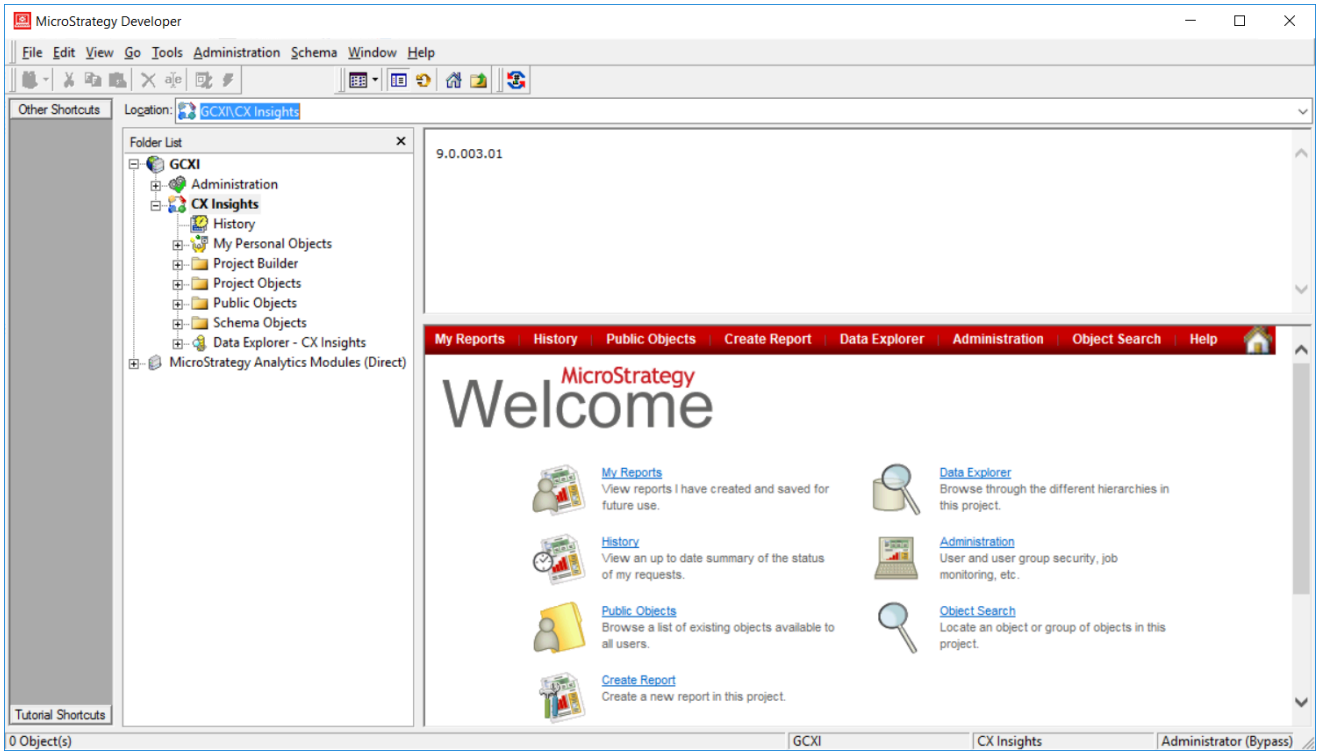
Genesys documentation, including this document, also supports mobile devices. In most cases, it easier to use Genesys CX Insights, and the documentation, in landscape mode.

MicroStrategy Developer

MicroStrategy Developer is the business intelligence software component you can use for viewing the definitions of universe elements and customizing metrics or dimensions, as well as for many other operation, administration, and maintenance activities.

For information on how to install and access MicroStrategy Developer, see the MicroStrategy ReadMe and MicroStrategy Installation documentation. Always install a MicroStrategy Developer release that matches the MicroStrategy Secure Enterprise Platform release (4-digit release number) provided in the GCXI container; for example, MicroStrategy Developer 11.2.0.x is compatible with GCXI releases that use MicroStrategy Secure Enterprise Platform 11.2.0.x.

Using MicroStrategy Developer



Use MicroStrategy Developer to view or edit the CX Insights Project, and to view or edit MicroStrategy metrics or dimensions.

1. Open MicroStrategy Developer from the location where you installed it. For example, on a typical Windows deployment, click **Start** > **Apps** > **MicroStrategy Products** > **Developer**.
2. When prompted, enter a valid user name and password. MicroStrategy Developer opens.
3. In the **Folder List**, navigate to the project folder, for example <**server name**> **CX Insights**.
4. Click **Click here to open project CX Insights.** A **Welcome** page appears, offering quick links to popular folders and actions.
5. To browse the reports and other objects that are provided out-of-box, click **Public Folders**.

For more information, including about how to safely customize reports or metrics, see the *Genesys CX Insights User's Guide* or search the MicroStrategy Product Documentation.

Checking Web Server Status

To check the status of the web server, visit the pages listed in the following table, where <server> is the name of the server where GCXI resides:

Visit this URL	To check this
http://<server>:8080/gcxi/monitor/gim	Check the status of the connection to the Info Mart database.
http://<server>:8080/gcxi/monitor/db	Check the status of the connection to the MicroStrategy meta database.
http://<server>:8080/gcxi/monitor/web	Check to see whether GCXI reports are available.

You should see output with values similar to the following examples. This example shows an error code indicating that there is no problems detected:

```
{
...
"Error code":0,
"Reason":"successfully verified databases: 1",
...
}
```

If you encounter other error codes, see the accompanying Reason field for an explanation, for example, if the connection to the database fails:

```
{
...
"Error code":1,
"Reason":"The connection attempt failed.",
...
}
```

Uninstalling Genesys CX Insights

Disclaimer

Genesys is committed to diversity, equality, and inclusivity. This includes using appropriate terms in our software and documentation. Therefore, Genesys is removing non-inclusive terms. For third-party products leveraged by Genesys that include such terms, Genesys uses the following as replacements.

- For the terms master/slave, Genesys uses “primary” and “secondary” or “primary” and “replica,” with exceptions for their use in third-party commands.
- For the terms blacklist/whitelist, Genesys uses blocklist/allowlist.
- For the term master, when used on its own, Genesys uses main wherever possible.

This page describes the steps needed to remove Genesys Customer Experience Insights (Genesys CX Insights) and supporting software, including Docker and Kubernetes.

Procedure: Uninstalling Genesys CX Insights and supporting software

Purpose: Use the steps in this procedure to uninstall Genesys CX Insights, and the software that supports it.

Steps

Caution: The following steps delete all Genesys CX Insights content, including reports and projects, and delete Genesys CX Insights meta db and related db users.

1. Choose one of the following options to delete the Genesys CX Insights meta and hist databases:
    - If you are using an external PostgreSQL server to host the meta database, execute the following command on the PRIMARY machine:

      ```
      kubectl apply -f <destination path>/gcxi-cleanup.yaml
      ```

      where <destination path> is the folder in which you deployed the software.

      OR
    - If you are using the pre-packaged meta database, execute the following command on the Control plane node machine:

```
kubectl delete -f <destination path>/gcxi-postgres.yaml
```

and delete the PostgreSQL data volume from the file system:

where <destination path> is the folder in which you deployed the software.

2. To reset Kubernetes , execute the following command, on each machine:

```
kubeadm reset
```

3. To clean up volumes (logs, pre-packaged PostgreSQL data, and the external data source cache), execute the following commands:
   1. On each machine:

      ```
      rm -rf /mnt/log
      ```

   2. On the Control plane node machine:

      ```
      rm -rf /genesys/gcxi/data
      ```

      This step removes any customizations you made to Genesys CX Insights.

   3. On each machine:

      ```
      rm -rf /genesys/gcxi/shared/*
      ```

      This step removes external datasources data.
4. To uninstall Kubernetes, see the documentation for Kubernetes website.
5. To uninstall Docker, see the documentation on the Docker website.

Upgrading Genesys CX Insights

This page describes how to upgrade your Genesys CX Insights deployment to a later release. For information about compatible releases, see the Product Alert. This page is applicable to deployments that use Kubernetes Descriptors. If you deployed using Helm, see Upgrade GCXI using Helm.

> **Important**
>
> In keeping with Genesys' commitment to diversity, equality, and inclusivity, beginning with release 9.0.019.01, some pod names are changed; this document refers to "gcxi-primary" and "gcxi-secondary" pods. In release 9.0.019.00 and earlier, these pods were named "gcxi-master" and "gcxi-slave".

If SAML is enabled in your environment, before you update GCXI, see the instructions in *Genesys CX Insights User's Guide* pertaining to Updating Genesys CX Insights when SAML is enabled.

Default user

Beginning with release 100.0.020.0000 (9.0.020), the default user for all Genesys CX Insights images has changed to **genesys (id = 500)**; previously, the default user was **root**. In scenarios where you upgrade from release 9.0.019 or earlier to release 9.0.020 or later, this can result in an error due to volume mount permission settings; to prevent this, you must perform one of the following:

- **Change ownership of volumes** — This is the recommended method; complete the following steps before you upgrade to release 9.0.020 or later
  1. Execute the following commands to stop the pods:

     ```
     kubectl scale deploy/gcxi-secondary --replicas=0
     kubectl scale deploy/gcxi-primary --replicas=0
     ```

  2. Execute the following commands to change user:group ownership of volumes (log, backup, shared):

     ```
     chown -R 500:500  /mnt/log
     chown -R 500:500  /genesys/gcxi/shared
     ```

  OR

- **Continue to run GCXI containers under user root.** — See the documentation that is appropriate to your container runtime:
  - Docker-compose (see user directive)
  - Kubernetes (see runAsUser directive)

After you have completed one of the preceding changes, proceed with Upgrading Genesys CX Insights from 9.0.x to a later 9.0.x release

Procedure: Upgrading Genesys CX Insights from 9.0.x to a later 9.0.x release

Steps

1. Obtain the latest GCXI image (for example, **gcxi:9.0.019.00**)
2. Execute the following command to retag the image:

   ```
   docker tag <repository name>/<image>:<release> <image>:<release>
   ```

   where

   <repository name> is the identifier for the repository from which you downloaded the files,

   <image> is the name of the image file, and <release> is the release number.

   For example:

   ```
   docker tag pureengage-docker-production.jfrog.io/gcxi:9.0.019.00 gcxi:9.0.019.00
   ```
3. Execute the following command to back up the GCXI meta db:

   ```
   kubectl apply -f <destination path>/gcxi-backup.yaml
   ```

   where <destination path> is the folder in which the Genesys Installation Package (IP) is stored.
4. Execute the following commands to stop the containers:

   ```
   kubectl scale deploy/gcxi-secondary --replicas=0

   kubectl scale deploy/gcxi-primary --replicas=0
   ```
5. Open the **gcxi.properties** file for editing, and change the value of **GCXI_VERSION** to match the release you are installing. For example:

   ```
   GCXI_VERSION=9.0.019.00
   ```
6. Execute the following commands to load **gcxi.properties** into Kubernetes:

   ```
   kubectl delete configmap gcxi-config

   kubectl create configmap gcxi-config --from-env-file=<destination
   path>/gcxi.properties --namespace genesys
   ```

   where <destination path> is the folder in which the Genesys Installation Package (IP) is stored.
7. Execute the following commands to update GCXI images in Kubernetes:

```
kubectl set image deployment/gcxi-<primary|secondary> gcxi=gcxi:<release>
```

Where:

- <release> is the same string as you entered for the GCXI_VERSION.

For example:

```
kubectl set image deployment/gcxi-primary gcxi=gcxi:9.0.019.00
```

```
kubectl set image deployment/gcxi-secondary gcxi=gcxi:9.0.019.00
```

8. Execute the following command to start the PRIMARY container:

```
kubectl scale deploy/gcxi-primary --replicas=1
```

Wait until PRIMARY container has started (Tomcat is up, and MicroStrategyWeb page is available).

9. Execute the following command to start the SECONDARY container:

```
kubectl scale deploy/gcxi-secondary --replicas=1
```

Upgrading the meta and history database

Use the following procedure to optionally upgrade the meta/hist database.

Procedure: Upgrading meta PostgreSQL

Purpose: If you have previously installed Genesys CX Insights with an older release of PostgreSQL hosting the meta and history database, you can optionally upgrade PostgreSQL using the steps in this procedure. The latest PostgreSQL release is required for new installations; when upgrading Genesys CX Insights you can continue to use the installed release of PostGreSQL for the meta and history database, but Genesys recommends that you upgrade PostgreSQL when you upgrade Genesys CX Insights.

Prerequisites

Follow the instructions in Upgrading Genesys CX Insights from 9.0.x to a later 9.0.x release to upgrade Genesys CX Insights before upgrading the PostgreSQL meta and history database.

Steps

1. Execute the following command to back up the GCXI meta db:

   ```
   kubectl apply -f <destination path>/gcxi-backup.yaml
   ```

   where <destination path> is the folder in which the Genesys IP is stored, for example:

   ```
   kubectl apply -f /genesys/gcxi/gcxi-backup.yaml
   ```

2. Execute the following commands to stop currently running containers:

   ```
   kubectl scale deploy/gcxi-secondary --replicas=0
   ```

   ```
   kubectl scale deploy/gcxi-primary --replicas=0
   ```

3. Execute the following command to delete the postgres container:

   ```
   kubectl delete -f <destination path>/gcxi-postgres.yaml
   ```

   where <destination path> is the folder in which the Genesys CX Insights is installed.

4. Move the local meta database folder (usually a subfolder of `/genesys/gcxi/data`) to a backup location. Once the upgrade is complete, you can delete this folder from the backup location; should the upgrade fail, you can restore this folder and restart the old postgres container.

5. Optionally, set the nodeSelector for the postgres pod; it is not set by default. If you do not have a shared volume set up for the **/data** folder (for example NFS), failing to set the nodeSelector can result in your meta database being stored locally on one one node, while the PostgreSQL schemas are stored on the other node:

   1. Open the **gcxi-postgres.yaml** file for editing.
   2. Uncomment the `## nodeSelector example ##` section, and edit it as described in the comments, so that **node-selector** is set to **worker node**. Ensure that the label in nodeSelector matches the one applied to the node.

6. From the IP of the updated release of Genseys CX Insights, get the **gcxi-postgres.yaml** file, open it for editing, and change the **image:** parameter to reflect the release of PostGreSQL you are installing.

7. Execute the following commands to start the postgres container using the new **gcxi-postgres.yaml** file:

   1. `kubectl create -f <destination path>/gcxi-postgres.yaml`

      where <destination path> is the folder in which the new Genesys IP is stored, for example:

      ```
      kubectl create -f /genesys/gcxi/gcxi-postgres.yaml
      ```

   2. Execute the following command to verify the state of the gcxi-postgres pod:

      ```
      kubectl get pods | grep 'gcxi-postgres*'
      ```

      The pod status should be `Running`, for example:

```
gcxi-postgres-5cd4d45754-mss6p   1/1       Running     0          6d
```

If it has any other state, wait a few minutes and check again (it may take some time).

8. Execute the following command to initialize the new postgres container:

```
kubectl apply -f <destination path>/gcxi-init.yaml
```

9. Restore the database using the two backup files (two **pg_dums mstr_meta** and two **mstr_hist** files):
   1. Examine the pg_dumps backup files, and copy the timestamp value. For example, if the file name is **mstr_hist...20210414-095919.pgdump**, copy the string `20210414-095919`.
   2. Open the **gcxi-restore.yaml** file for editing, and populate the RESTORE_TAG parameter using the timestamp value you saved from the pgdump files. For example:
   3.
```
- name: RESTORE_TAG
value: 20210414-095919
```
   4. Execute the following command to restore the meta and history database:

```
kubectl apply -f <destination path>/gcxi-restore.yaml
```

   5. Execute the following command and verify that the `gcxi-restore-*` job is `Completed`:

```
kubectl get pod
```

10. Start your GCXI pods by start both PRIMARY and SECONDARY pods:

```
kubectl scale deploy/gcxi-primary --replicas=1
```

Wait until PRIMARY is done (wait until Tomcat is up, and MicroStrategyWeb page is available).

11. Execute the following command to start the SECONDARY container:

```
kubectl scale deploy/gcxi-secondary --replicas=1
```

Best practices when adopting Genesys CX Insights

If you are adopting Genesys CX Insights into your environment for the first time, use the information on this page to learn more about what practices Genesys recommends, and what you can change in the MicroStrategy projects and what you cannot. This page applies to all Genesys CX Insights projects (including, for example, Genesys CX Insights for iWD). This page requires an understanding of MicroStrategy; for more information about MicroStrategy, see the Additional resources page.

In most respects, the “CX Insights” project is an ordinary MicroStrategy project, so in many cases, you can modify it as you would any other MicroStrategy project. This page provides additional detail about what you can modify without damaging the project.

> **Important**
>
> Failing to follow the guidelines on this page can damage your Genesys CX Insights deployment, and could require that you redeploy the software. If you plan to experiment with changes other than those outlined on this page, Genesys recommends that you do so with a project other than a live Genesys CX Insights project.

Extending an Existing MicroStrategy Project

This section distinguishes between custom objects (objects created specifically for your organization), and objects that are shipped as part of the Genesys CX Insights project. In most cases, you will modify only custom objects, which are referred to as "objects you created" throughout this document.

Creating new objects

You can create new objects. When you do so, always follow these two rules:

- When you create an object, always save it in a folder you create, except in the following circumstances:
  - You can safely create objects of the type "folder" within **CX Insights** folders.
  - You can safely create any supported object within the **CX Insights** > "Custom" folder.
- Some objects can be saved only in specific locations; for example: users, user groups, security roles, schedules. If you create objects of these types, you must save them in the same locations where **CX Insights** objects of the same type are stored. It's important that you give them names that makes it easy to distinguish from the **CX Insights** objects.

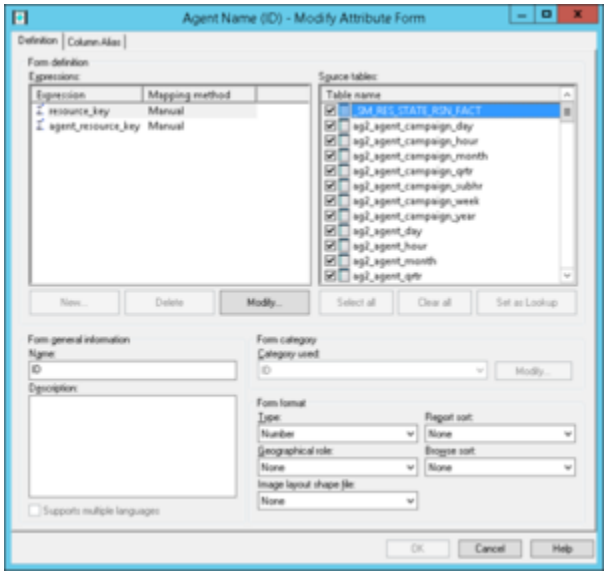
Changing existing objects

You can change any objects you have created without restrictions. This section discusses limitations on the changes you can make to the default **CX Insights** objects.

In most cases, you must avoid making changes to **CX Insights** objects, except as described in the following three sections:

- Exception 1: Attribute Forms
- Exception 2: User Groups
- Exception 3: Users

In all three cases, it's important to note that you can safely change *only the properties specifically named*; avoid changing anything that is not mentioned. To make these changes, ensure that you have access to the the necessary software (see Accessing CX Insights GUIs) and the appropriate permissions (talk to your system administrator).



Modify Attribute Form dialog in MicroStrategy Developer

Exception 1: Attribute Forms

You can modify the *forms* of the attributes as follows:

- Add new expressions to existing Attribute Forms.
- Add new source tables to existing expressions.

To modify attribute forms, use the Modify Attribute Form dialog in MicroStrategy Developer as shown in the figure **Modify Attribute Form dialog in MicroStrategy Developer**. Modify or delete only those expressions and source tables you added; do not change other attribute properties nor add new Attribute Forms.

Exception 2: User Groups

You can safely modify the default user groups as follows:

- You can change a user group permissions on your project(s) and on your object(s).
- You can change the list of users included in a group.

Exception 3: Users

You can safely create, modify, and delete your own users. You can safely modify any properties of the default "CX Insights" users, but you **must not** change:

- Permissions assigned to a “CX Insights” user on “CX Insights” project, or on “CX Insights” objects.
- The name of the CX Insights user.

For more information about users and user groups, see Managing Users, Groups, and Privileges in the *Genesys CX Insights User's Guide*.

Deleting objects

You can delete objects you create. You must not delete the “CX Insights” objects.

Renaming and moving objects

You can rename or move objects you create. You must not rename “CX Insights” objects or folders, and you must not move them into different folders.

Typical workflow

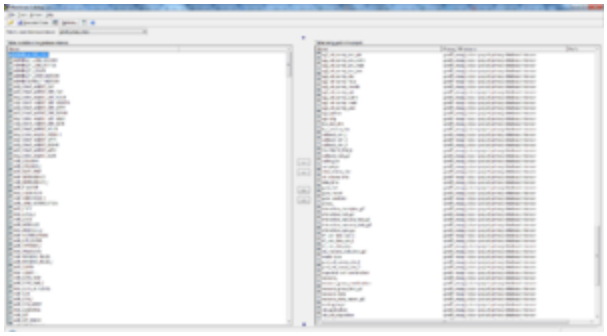
This section provides a high-level walkthrough of the steps you might follow to modify the Agent Name attribute, which is located in the **CX Insights\GCXI\Agent** folder.

1. Data source preparation

Before you add or change the project metadata, always prepare or change the data source on which the project depends, so that the data source contains all required objects.

Genesys recommends that you follow these conventions:

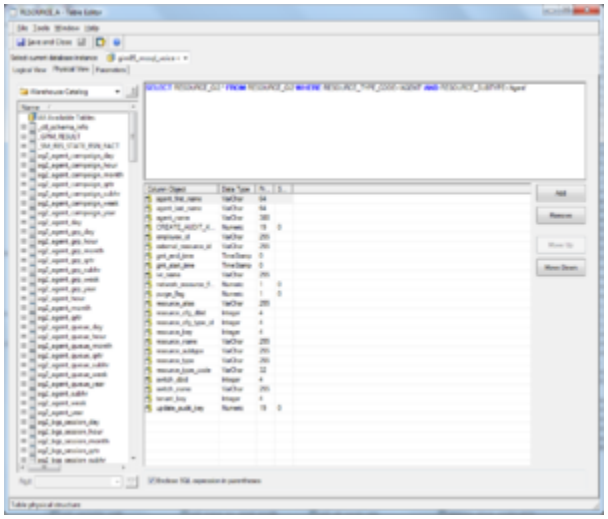
- You always create names for the tables standing for aggregation sets that begin with the "ag2_" prefix.
- You indicate the aggregation level of a table by an appropriate suffix ("_day", "_subhr", etc.).
- You always include the word FACT in the names of fact tables.



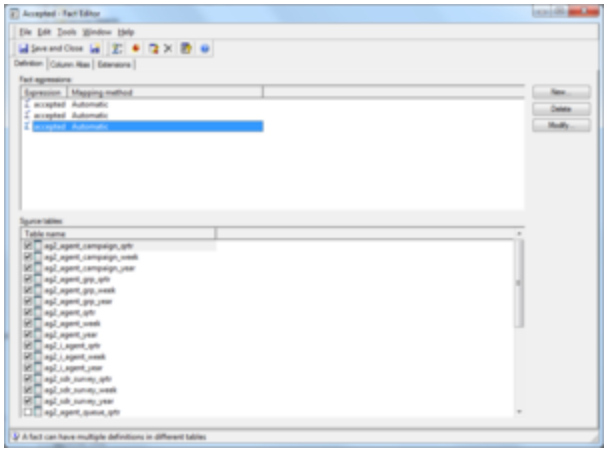
Example: all agents information stored in RESOURCE_ GIM database table

2. CX insights Warehouse

To add new tables that have a direct equivalent in the database, in MicroStrategy Developer, choose "Schema" > "Warehouse Catalog". When you do so, the application queries the database, and presents a list of tables and their columns to import into the MicroStrategy schema model. From the list, select the tables you want to add to the project. See the figure **Example: all agents information stored in RESOURCE_ GIM database table**. For more information, see Data warehouse and project interaction: Warehouse Catalog.



Example: Logical view RESOURCE_A, created to filter agent types resources from other types



Example: Scheme Objects\Facts\Accepted

3. CX Insights\Schema Objects

This section describes three Scheme Objects subfolders:

- **Scheme Objects\Tables**:
  MicroStrategy also enables you to create Logical Views that resemble relational database views. Views are named row sets based on SQL statements. Unlike tables, the column list of a view must be edited manually. In other respects, views behave in the same way as tables.

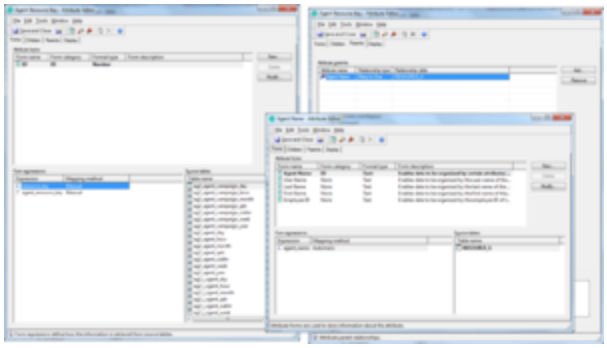
  Genesys recommends that you always create such objects in the **Schema Objects\Tables\Custom** folder. See the figure **Example: Logical view RESOURCE_A, created to filter agent types resources from other types**.

- **Scheme Objects\Facts**:
  Facts form the basis for metrics, which are used in the majority of analyses and reports that users can create. The **Scheme Objects\Facts** folder and its subfolders contain facts, which are divided by the subfolders into themes. The root **Scheme Objects\Facts** folder contains facts participating in more than one aggregation set and sometimes in fact tables, while subfolders contain facts for individual thematic aggregates, or for fact tables, or both. See the figure **Example: Scheme Objects\Facts\Accepted**.

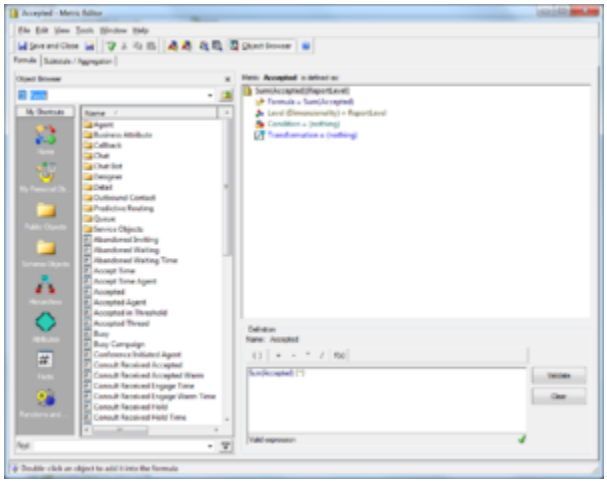
  Genesys recommends that you always create such objects in the “Schema Objects\Facts\Custom” folder.

  For more information, see Project Design and Facts.

- **Scheme Objects\Hierarchies**
  You can save custom hierarchies in the “Schema Objects\Hierarchies\Data Explorer” project folder. Do not modify existing CX Insights hierarchies.



Example: CX Insights\GCXI\Agent\Agent Resource Key and Agent Name attributes



Example: CX Insights\GCXI\Agent\Activity\Accepted

4. CX Insights\GCXI

This folder contains several subfolders:

- **Attributes**
  The Attributes folder contains many attributes objects that can be used in more than one report. If making changes to objects in this folder, be sure that your changes are compatible with other attributes in the GCXI folder. The easiest way to verify this, is to examine SQL expressions in your final reports, and ensure that the attribute in the MicroStrategy engine is an element that provides JOINs to the FROM clause of SQL expressions. See the figure **Example: CX Insights\GCXI\Agent\ Agent Resource Key and Agent Name attributes**.

  For more information, see the MicroStrategy Project Design Guide.

- **Metrics/Filters/Prompts/Reports**
  Objects of these types cannot be re-used, and must be created as new objects. Objects of these types are placed in folders, and subfolders, organized by theme. Genesys recommends that you observe this practice by creating thematic folders and subfolders, or use the Custom folders. See the figure **Example: CX Insights\GCXI\Agent\Activity\Accepted**.

  For more information, see the the following MicroStrategy resources:

  - Metrics
  - Prompts
  - Filter Data

- **Dashboards**
  You can create dashboards only in the MicroStrategy Web interface. For more information, see Dashboards and Dossiers.

5. Others

There are three conditional groups of users who can make changes to the CX Insights project:

- Contributors — Can complete all workflow steps.
- Genesys Premise Customers — Can complete all workflow steps.
- Genesys Engage cloud Customers — Can work with all objects described in 4. CX Insights\GCXI, **except** Attributes objects.

Troubleshooting

This page provides some common troubleshooting tips. Many topics on this page require advanced knowledge. For more information or assistance, talk to your Genesys representative, and see the links at the bottom of this page.

> **Important**
>
> In keeping with Genesys' commitment to diversity, equality, and inclusivity, beginning with release 9.0.019.01, some pod names are changed; this document refers to "gcxi-primary" and "gcxi-secondary" pods. In release 9.0.019.00 and earlier, these pods were named "gcxi-master" and "gcxi-slave".

Container startup logic

This section describes the general logic of GCXI startup. Understanding this process can help you to troubleshoot problems.

About gcxi-init

You execute this job when GCXI is first installed, at which time it:

1. creates GCXI service objects in the specified database server: meta / history databases and logins.
2. loads gcxi data (initial Projects) into newly created databases.

Once gcxi-init completes, we expect that the Microstrategy metadata database (the "meta db") has been created. You can perform the following actions to verify that this is the case:

1. Check the gcxi-init logs to ensure that the Microstrategy metadata database has been created without error.
2. Check DB Server to ensure that the meta database is available.

Main gcxi container startup

Each time the gcxi container starts, it:

1. Checks the environment variables and default values to determine the MicroStrategy Administrator password, and optionally changes the password.
2. Starts MicroStrategy server inside the container (a valid meta db is required for MicroStrategy to start properly).
   If you encounter a problem at this stage, check the following:
   - if MSTR server fails to start immediately, ensure that the meta db is available.
   - if MSTR server fails to start after a delay of about 3 minutes, and exits with the error `Failed to start MSTR server!`, check sysctls / shared memory settings on the host.
3. Runs Command Manager (a Microstrategy client tool) using the MicroStrategy Administrator account and password, and executes the following commands in Command Manager:

   ```
   LIST ALL DBLOGINS;

   LIST ALL DBCONNECTIONS;

   LIST ALL DBINSTANCES;
   ```

   If these commands fail, ensure that Command Manager is able to log in to the MicroStrategy server (ensure that a valid MicroStrategy Administrator password was provided in step 1).
4. Creates DSNs (MicroStrategy connectivity objects), which are used to connect to Genesys Info Mart and other data sources. The DSNs are created based on the container DSNDEF* environment variable.
5. If the value of the environment variable GCXI_VERSION is later than the current GCXI release number, an upgrade procedure starts. This one-time task can take up to 30 minutes to complete.
6. Performs other MicroStrategy server configurations (some are executed only by a PRIMARY container).
7. Configures and starts Tomcat.

Out of Memory error in log when docker-compose -f docker-compose.yml up is executed on

Windows

When starting GCXI on Windows, an error can appear if there is not sufficient memory configured for the virtual machine. Resolve this as follows:

1. Open Hyper-V Manager.
2. Open the **Settings** of your virtual machine, and navigate to the section **Hardware** > **Memory**.
3. Select the option **Enable Dynamic Memory**.
4. In the Windows system tray, access the Docker context menu, and open **Settings** of Docker Desktop “Settings”.
5. In the section **Resources** > **Advanced**, increase the allocated memory, and save your changes.

Port error when compose runs on Windows

The following error can appear when you start Docker under Windows:

```
>docker-compose -f docker-compose.yml up
<…>
ERROR: for gcxi-0  Cannot start service gcxi-0: Ports are not available: listen tcp 0.0.0.0:8080: bind: Only one usage of each socket
address (protocol/network address/port) is normally permitted.
ERROR: Encountered errors while bringing up the project.
```

This indicates that the port 8080 is in use. To resolve this issue:

1. Open the *docker-compose.yml* file for editing.
2. Change the port mapping as follows: edit the line **8080:8080**, changing the first value to **<unused_ port_in_windows>:8080”, where <unused_ port_in_windows> is an unused port, for example** 8280:8080**. If you remap this port, be sure to use the new port value when accessing MicroStrategy web interface.**

Could not translate host name "gcxi-postgres" to address: Name or service not known

In some scenarios, an error can appear when you create the MicroStrategy meta database, similar to the following:

```
++ psql --host gcxi-postgres --port 5432 --username postgres --dbname postgres -tAc 'SELECT 1'
psql: could not translate host name "gcxi-postgres" to address: Name or service not known
++ '[' 2 -eq 0 ']'
++ code=1
++ echo 'Connection to meta db server failed. Either wrong admin creds provided, or DB is broken.'
```

To work around this issue:

1. Open the **gcxi-init.yaml** file for editing, and find this line:

   ```
   command: [ "/bin/sh", "-c", " ${GCXI_INSTALL_HOME}/run_init.sh" ]
   ```

2. Edit it to read:

   ```
   command: [ "/bin/sh", "-c", "echo '<GCXIpostgresIP> gcxi-postgres' >> /etc/hosts && ${GCXI_INSTALL_HOME}/run_init.sh" ]
   ```

   Where <GCXIpostgresIP> is the IP address of the gcxi-postgres pod, for example:

   ```
   command: [ "/bin/sh", "-c", "echo '10.244.0.4 gcxi-postgres' >> /etc/hosts && ${GCXI_INSTALL_HOME}/run_init.sh" ]
   ```

3. Try again to create the MicroStrategy meta database.

Genesys CX Insights fails after restart

If Genesys CX Insights fails after restarting, one possible cause is a missing/disabled Project, which prevents the GCXI container from starting. Check to ensure that required Projects in the meta database are not in a disabled state:

1. Open MicroStrategy Developer.
2. Right-click the server name, and select **Configure MicroStrategy Intelligence Server**.

3. In the **Project** > **General** section, ensure that check boxes are selected next to required Project names.

Container Diagnostics

If the GCXI container fails on startup, and the reason for the failure is not immediately evident, Genesys recommends that you set the following debugging variables, and re-run the GCXI main container:

- `DEV_XTRACE_LEVEL=ALL`
  This provides super-extended logging, which is helpful in many scenarios.

  > **Warning**
  > Warning: all passwords are visible in container logs. Genesys recommends that you do not share this log with Customer Care, or any other party, unless you have first removed all sensitive information, such as passwords, from the log file.

- `DEV_ERR_EXIT=false`
  This provides more information in scenarios where the container does not exit on error and continues to run.

Generating logs

If you need assistance from Genesys, you may be asked to provide logging output. At a minimum, provide the logs, saving each output to a separate file with a name that clearly identifies each one. Note that this example is for Kubernetes; if you use a container engine other than Docker, use corresponding commands for that engine:

```
## general docker and k8s inspection
kubectl version
docker version
docker info
```

```
docker images
service docker status

## kubernetes objects
 # please provide the FULL log
kubectl log <gcxi_primary_pod>
kubectl get pod --all-namespaces -o wide
kubectl get node -o wide
kubectl get events --all-namespaces
kubectl get job -o wide

kubectl get cm gcxi-config -o yaml
kubectl get job gcxi-init -o yaml
kubectl get deploy gcxi-primary -o yaml
kubectl get pod <gcxi_primary_pod> -o yaml

# is containerized postgres used
kubectl log <gcxi_postgres_pod>
kubectl get pod <gcxi_postgres_pod> -o yaml

# if gcxi-init job instance still exists
kubectl log <gcxi_init_pod>
kubectl get pod <gcxi_init_pod> -o yaml

## linux info
hostnamectl
free -h
df -h
sestatus

## sysctls
sysctl kernel.sem
sysctl net.bridge
sysctl vm
```

Kubernetes Inspection

If you suspect that Kubertnetes might be the source of a problem, see the document https://learnk8s.io/troubleshooting-deployments, which provides a troubleshooting flow-chart to guide you through troubleshooting Kubernetes in your deployment.

MicroStrategy resources

This section provides links that are useful when you are performing troubleshooting. The URLs in the following table point to the latest page on the MicroStrategy documentation site. If the documentation page that loads is more recent than your installed release, you can modify the URL by replacing **Current** with a specific release number.

Topic	URL
What's New	https://www2.microstrategy.com/producthelp/Current/Readme/en-us/Content/whats_new.htm
System Settings / IPC Requirements	https://www2.microstrategy.com/producthelp/Current/InstallConfig/en-us/Content/Software_requirements_and_recommendations.htm
Advanced Reporting Guide	https://www2.microstrategy.com/producthelp/Current/AdvancedReportingGuide/WebHelp/Lang_1033/Content/home.htm
MSTR Web Guide	https://www2.microstrategy.com/producthelp/current/MSTRWeb/WebHelp/Lang_1033/Content/home_mstrweb.htm
MSTR Supported Configurations	https://www2.microstrategy.com/producthelp/Current/Readme/en-us/Content/certified_configurations.htm Click **View Dossier** to learn more about certified product releases.
Search for other topics	You can easily search for detailed information about MicroStrategy products: 1. On the MicroStrategy Community Search Page, enter your search terms. 2. Filter your search results by selecting the Document Version (such as **2020**).

Additional resources

The following resources provide additional information that is relevant to this software. Consult these additional resources, as necessary.

Genesys CX Insights

Documentation for Genesys Customer Experience Insights (CX Insights) is available on the Genesys Documentation website:

- *Genesys CX Insights Deployment Guide*, which will help you install, start, stop, and uninstall the Genesys-provided image of MicroStrategy and the CX Insights Project and reports.
- *Genesys CX Insights User's Guide*, which includes a report- customization example that displays aggregated results that are sectioned by your own custom user data.
- *Genesys CX Insights Projects Reference Guide*, which describes objects that are used in Genesys CX Insights projects and reports, focusing on metrics, attributes, and the folders that are used to organize them.
- *Genesys CX Insights Hardware Sizing Guide*, which provides information about hardware sizing for typical contact center scenarios.
- Genesys CX Insights Release Notes, Product Alerts, and What's New are available on the GCXI page of the Genesys documentation site.

MicroStrategy

Documentation for MicroStrategy software is available on the MicroStrategy Learning Center or Help page, or in an electronic format that you can download to your mobile device (QR codes).

Easy search for MicroStrategy topics

- MicroStrategy Community Search Page

> **Tip**
> On the Community Search Page, filter your search results by selecting the Document Version (such as **2020**).

Following are some popular topics, and where to find information about them on the MicroStrategy Wiki:

The latest information from MicroStrategy

- What's New in MicroStrategy
- Key information about MicroStrategy Web
- Key information about MicroStrategy Developer

Analyzing data in a MicroStrategy report or dashboard

- Basic Reporting Guide
- Mobile Analysis Guide

Creating dashboards and reports

- Enterprise Reporting
  - Document Creation Guide
  - Dashboard and Widgets Guide
- Slice and Dice Analysis
  - Basic Reporting Guide
  - Advanced Reporting Guide
- Advanced and Predictive Analysis
  - Advanced Reporting Guide
  - Function Reference Guide
- Alerts and Proactive Notification
  - System Administration Guide
  - Mobile Analysis Guide
- OLAP Analysis
  - In-memory Analytics Guide
- Integrate data reporting with Microsoft Office
  - MicroStrategy Office User Guide

Installing or upgrading MicroStrategy

- Installation and Configuration Guide
- Upgrade Guide

Modelling your data and designing a project

- Project Design Guide

- MDX Cube Reporting Guide

Configuring and Administering MicroStrategy

- System Administration Guide
- Timeout settings in MicroStrategy Web
- User Session Idle Timeout

MicroStrategy Quick Start

- Quick Start Guide

Docker

- About Docker

Kubernetes Installation

- Kubernetes Getting Started
- Installing kubeadm

OpenShift

- OpenShift documentation

Helm

- Helm documentation

Genesys Info Mart

Documentation for Genesys Info Mart is available on the Genesys Documentation website:

- *Genesys Info Mart Operations Guide*, for information about Genesys Info Mart jobs such as `Job_AggregateGIM` and the Genesys Info Mart Manager for managing Genesys Info Mart jobs.
- *Genesys Info Mart Deployment Guide*, for information about configuring the Genesys Info Mart and Interaction Concentrator servers to recognize user data.

Reporting and Analytics Aggregates

Documentation for Reporting and Analytics Aggregates (RAA) is available on the Genesys Documentation website:

- *Reporting and Analytics Aggregates Deployment Guide*, which describes the runtime parameters and configuration options mentioned in this document.
- *Reporting and Analytics Aggregates User's Guide*, which describes the different modes of running aggregation, the aggregation hierarchies, and how to configure Reporting and Analytics Aggregates (RAA) to aggregate data based on these user-defined dimensions.
- The Physical Data Model documentation for your RDBMS, which describes the aggregate tables and subject areas:
  - *Reporting and Analytics Aggregates Physical Data Model for a Microsoft SQL Server Database*
  - *Reporting and Analytics Aggregates Physical Data Model for an Oracle Database*
  - *Reporting and Analytics Aggregates Physical Data Model for a PostgreSQL Database*

Genesys

Additional documentation for Genesys products is available, as follows:

- The Genesys Glossary provides a comprehensive list of the Genesys and computer-telephony integration (CTI) terminology and acronyms.
- *Genesys Migration Guide*, available on the Genesys Documentation website, provides documented migration strategies for Genesys product releases. Contact Genesys Customer Care for more information.
- Release Notes and Product Advisories for each Genesys product, which are available on the Genesys Documentation website.

Information about supported hardware and third-party software is available on the Genesys Documentation website in the following documents:

- The Genesys CX Insights page in the *Genesys Supported Operating Environment Reference Guide*
- *Genesys Supported Media Interfaces Reference Manual*
- *Genesys Hardware Sizing Guide*, which provides information about Genesys hardware sizing guidelines for the Genesys 8.x releases. For additional system-wide planning tools and information, see the release-specific listings of System-Level Documents on the Genesys Documentation website (docs.genesys.com).

Other Genesys product documentation is available on the:

- Genesys My Support website (formerly Customer Care)
- Genesys Documentation website
- Genesys Documentation Library DVD, which you can order by email from Genesys Order Management at Genesys Order Management.