Genesys Info Mart Private Edition Guide

5/10/2024

Table of Contents

Contents

Find links to all the topics in this guide.

Related documentation:

-
-
-

RSS:

- For private edition

Genesys Info Mart is the backend service behind historical reporting in the Genesys Multicloud CX private edition offering.

Overview

Learn more about Genesys Info Mart, its architecture, and how to support high availability and disaster recovery.

- About Genesys Info Mart
- Architecture
- High availability and disaster recovery

Configure and deploy

Find out how to configure and deploy the services in the Genesys Info Mart service group:

- GIM Stream Processor (GSP)
- GIM Config Adapter (GCA)
- GIM and the Info Mart database
- Before you begin GSP deployment
- Configure GSP
- GSP configuration options
- Deploy GIM Stream Processor

- Before you begin GIM deployment
- Configure GIM
- GIM configuration options
- Deploy GIM
- Before you begin GCA deployment
- Configure GCA
- Deploy GIM Config Adapter

Upgrade, roll back, or uninstall

Find out how to upgrade, roll back, or uninstall the Genesys Info Mart services and to migrate the Info Mart database.

- Upgrade, roll back, or uninstall Genesys Info Mart

Observability

Learn how to monitor Genesys Info Mart with metrics and logging.

- Observability in Genesys Info Mart
- GSP metrics and alerts
- GCA metrics and alerts
- GIM metrics and alerts

Info Mart database

Learn about the Info Mart database.

- Genesys Info Mart Historical Database Reference

About Genesys Info Mart

Contents

Learn about Genesys Info Mart and how it works in Genesys Multicloud CX private edition.

Related documentation:

-
-
-
-

RSS:

- For private edition

Genesys Info Mart is the enterprise-level Genesys service behind the historical reports in your cloud deployment. Genesys Info Mart receives data from various upstream Genesys services, then processes the low-level data to produce a data mart that the Genesys Multicloud CX historical reporting presentation layer, called Genesys CX Insights (GCXI), uses for contact center historical reporting.

Genesys Info Mart comprises three services—GIM Stream Processor (GSP), GIM Config Adapter (GCA), and Genesys Info Mart (GIM). Upstream services responsible for managing contact center configuration, interaction activity, agent activity, and so on publish messages to Apache Kafka. Genesys Info Mart consumes the data in Kafka reporting topics and, on a regular ETL cycle, transforms the data into a form more suitable for data analysis, then loads the data into the Info Mart database.

The Info Mart database stores the processed data. In addition, a separate aggregation service called Reporting and Analytics Aggregates (RAA) aggregates or re-aggregates the processed data and stores the aggregate data in the Info Mart database. The historical reports available in your cloud deployment are based on this aggregate data.

Supported Kubernetes platforms

Genesys Info Mart is supported on the following cloud platforms:

- Google Kubernetes Engine (GKE)
- Amazon Kubernetes Service (AKS)

See the Genesys Info Mart Release Notes for information about when support was introduced.

Features and functionality

Content coming soon

Architecture

Contents

Learn about Genesys Info Mart architecture

Related documentation:

-
-
-
-

RSS:

- For private edition

Introduction

Genesys Info Mart is the historical reporting back-end service that performs extract, transform, and load (ETL) processing of data generated by contact center activity. Genesys Info Mart comprises the following services:

- GIM Config Adapter (GCA) — Reads configuration data from the contact center Configuration Server database and stores the data into Kafka topics for consumption during transformation processing. There is a separate GCA service for each tenant.
- GIM Stream Processor (GSP) — Processes GCA-sourced configuration data as well as raw event data streamed to Kafka from upstream services handling interaction and agent activity on Voice and Digital channels. GSP transforms the data into data structures suitable for a data mart for contact center historical reporting.
- GIM — Pulls the transformed data from Kafka and loads this data into the Info Mart database. The GIM service also performs other database-related tasks, such as a regular maintenance job and an export job to export Info Mart data. There is a separate GIM service and Info Mart database for each tenant.

All the Genesys Info Mart services are stateful. This has implications for high availability (HA) and disaster recovery (DR) (see High availability and disaster recovery) as well as for upgrade methods, because you cannot deploy mirrored services in parallel.

For information about the overall architecture of Genesys Multicloud CX private edition, see the high-level Architecture page.

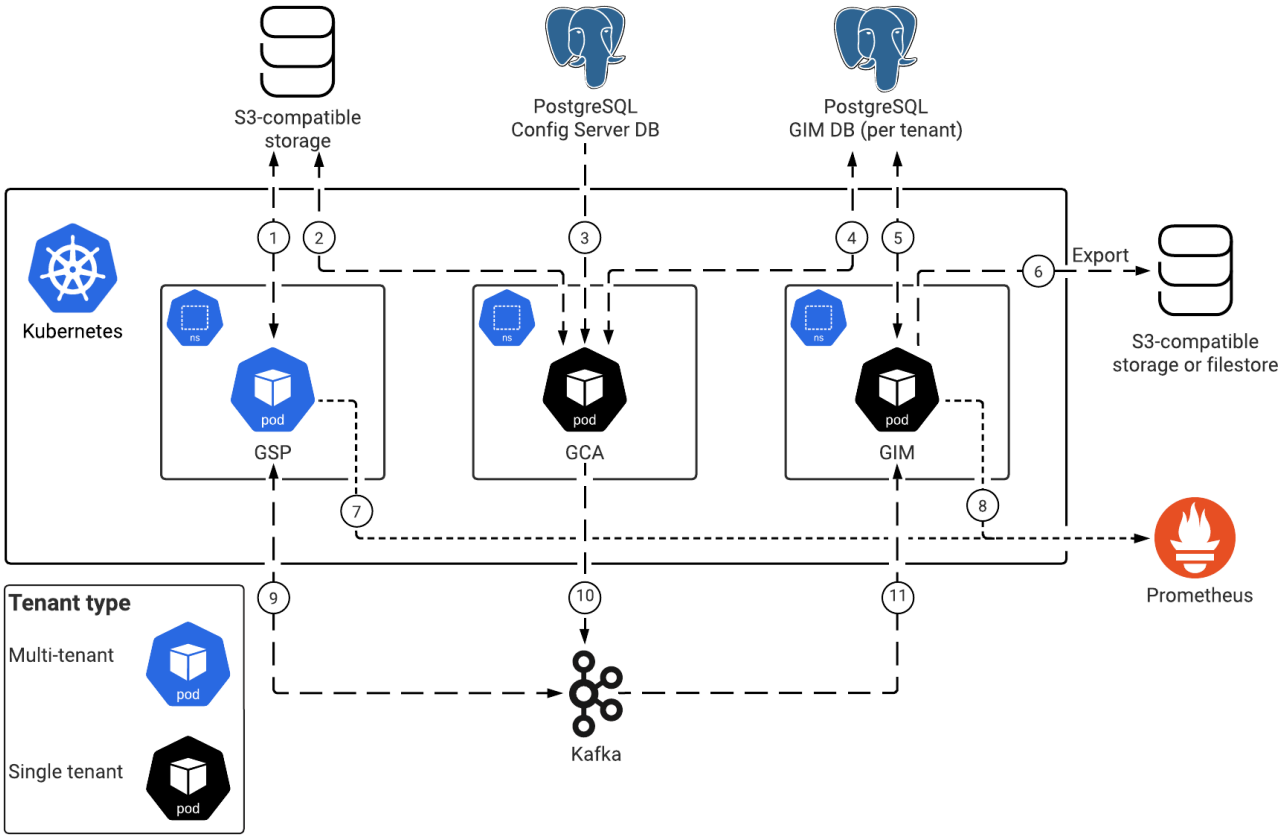
The following diagrams show the Genesys Info Mart architecture.

- Architecture diagram — Data flows is from the point of view of data.
- Architecture diagram — Connections is from the point of view of network connections.

See also High availability and disaster recovery for information about high availability/disaster recovery architecture.

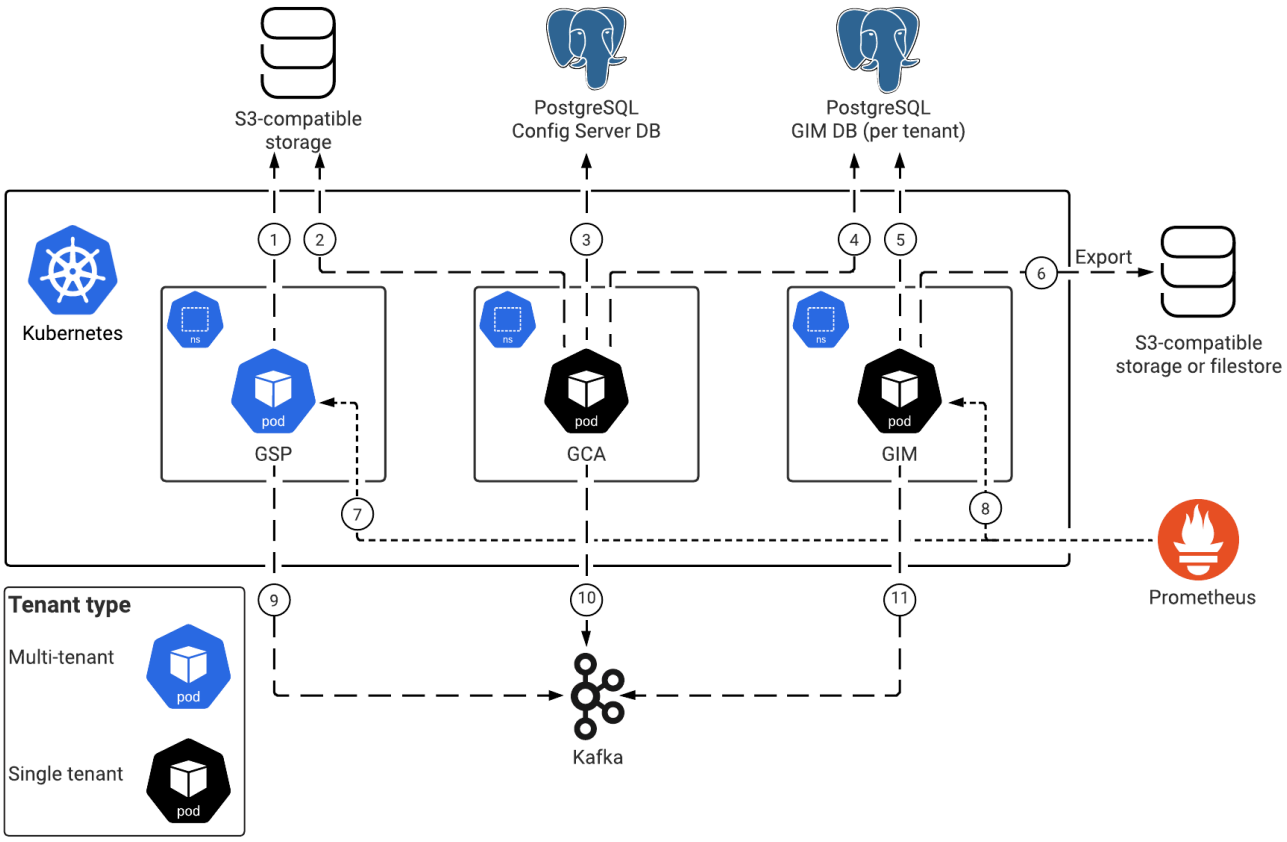
Architecture diagram — Data flows

The numbers on the connection lines refer to the connection numbers in the table that follows the diagrams. The direction of the arrows indicates the direction of data flows.



Architecture diagram — Connections

The numbers on the connection lines refer to the connection numbers in the table that follows the diagrams. The direction of the arrows indicates where the connection is initiated (the source) and where an initiated connection connects to (the destination), from the point of view of Genesys Info Mart as a service in the network.



Connections table

The connection numbers refer to the numbers on the connection lines in the diagrams. The **Source**, **Destination**, and **Connection Classification** columns in the table relate to the direction of the arrows in the Connections diagram above: The source is where the connection is initiated, and the destination is where an initiated connection connects to, from the point of view of Genesys Info Mart as a service in the network. *Egress* means the Genesys Info Mart service is the source, and *Ingress* means the Genesys Info Mart service is the destination. *Intra-cluster* means the connection is between services in the cluster.

Connection	Source	Destination	Protocol	Port	Classification	Data that travels on this connection
1	GIM Stream Processor	Object storage	HTTPS	443	Egress	GSP state information, checkpoints, and other data used during processing

Connection	Source	Destination	Protocol	Port	Classification	Data that travels on this connection
2	GIM Config Adapter	Object storage	HTTPS	443	Egress	Configuration data stored as a GCA snapshot used during processing
3	GIM Config Adapter	Configuration Database	TCP	5432	Egress	Contact center configuration data, which GCA uses to construct the GCA snapshot used during processing
4	GIM Config Adapter	Info Mart database	TCP	5432	Egress	Configuration data to synchronize the GCA snapshot with configuration data stored in the Info Mart database
5	GIM	Info Mart database	SSL	5432	Egress	Transformed reporting data to be stored in the Info Mart database. GIM also uses this connection to perform database maintenance.
6	GIM	Object storage	HTTP	443	Egress	Reporting data exported from the Info Mart database
7	Prometheus	GIM Stream Processor	HTTP	9249	Ingress	Metrics for monitoring and alerting
8	Prometheus	GIM	HTTP	8249	Ingress	Metrics for monitoring

Connection	Source	Destination	Protocol	Port	Classification	Data that travels on this connection
						and alerting
9	GIM Stream Processor	Kafka	Kafka	9092	Egress	Message bus for receiving and sending reporting data
10	GIM Config Adapter	Kafka	Kafka	9092	Egress	Message bus for sending configuration data
11	GIM	Kafka	Kafka	9092	Egress	Message bus for receiving transformed reporting data

Ports used for internal purposes

GSP uses the following ports for internal connections with Flink.

Port	Protocol
Task manager	
6122	RPC
Job manager	
6123	RPC
6124	Blob
8081	REST

High availability and disaster recovery

Find out how this service provides disaster recovery in the event the service goes down.

Related documentation:

-
-
-
-

RSS:

- For private edition

Service	High Availability	Disaster Recovery	Where can you host this service?
Genesys Info Mart	N = 1 (singleton)	Limited active spare	Primary or secondary unit

See High Availability information for all services: High availability and disaster recovery

Content coming soon

Before you begin GSP deployment

Contents

Find out what to do before deploying GIM Stream Processor (GSP).

Related documentation:

-
-
-
-

RSS:

- For private edition

Limitations and assumptions

Not applicable

Download the Helm charts

To configure and deploy the GIM Stream Processor (GSP), you must obtain the Helm charts included with the GSP release. These Helm charts provision GSP plus any Kubernetes infrastructure GSP requires to run.

GSP is the only service that runs in the GSP container.

For information about how to download the Helm charts, see Downloading your Genesys Multicloud CX containers. To find the correct Helm chart version for your release, see Helm charts and containers for Genesys Info Mart for the Helm chart version you must download for your release.

Third-party prerequisites

For information about setting up your Genesys Multicloud CX private edition platform, see Software requirements.

The following table lists the third-party prerequisites for GSP.

Third-party services

Name	Version	Purpose	Notes
Kafka	2.x	Message bus.	The Kafka topics that GSP will consume and

Name	Version	Purpose	Notes
			produce must exist in the Kafka configuration. See more details below.
Object storage		Persistent or shared data storage, such as Amazon S3, Azure Blob Storage, or Google Cloud Storage.	Both GSP and GCA require persistent storage to store data during processing. You can use the same storage account for both services.
A container image registry and Helm chart repository		Used for downloading Genesys containers and Helm charts into the customer's repository to support a CI/CD pipeline. You can use any Docker OCI compliant registry.	
Command Line Interface		The command line interface tools to log in and work with the Kubernetes clusters.	

Storage requirements

GSP maintains internal “state,” such as such as GSP checkpoints, savepoints, and high availability data, which must be persisted (stored). When it starts, GSP reads its state from storage, which allows it to continue processing data without reading data from Kafka topics from the start. GSP periodically updates its stare as it processing incoming data. GSP uses S3-compatible storage to store persisted data. You must provision this S3-compatible storage in your environment.

By default, GSP is configured to use Azure Blob Storage, but you can also use S3-compatible storage provided by other cloud platforms. Genesys expects you to use the same storage account for GSP and GCA.

Create S3-compatible storage

Genesys Info Mart has no special requirements for the storage buckets you create. Follow the instructions provided by the storage service provider of your choice to create the S3-compatible storage.

- For GKE, see the Google Cloud Storage documentation about creating bucket storage.
- For AKS, see Azure Storage documentation about creating blob storage.

To enable the S3-compatible storage object, you populate Helm chart override values for the service (see). To do this, you need to know details such as the endpoint information, access key, and secret.

> **Important**
> Note and securely store the bucket details, particularly the access key and secret, when you create the storage bucket. Depending on the cloud storage service you choose, you may not be able to recover this information subsequently.

Network requirements

No special network requirements. Network bandwidth must be sufficient to handle the volume of data to be transferred into and out of Kafka.

Browser requirements

Not applicable

Genesys dependencies

There are no strict dependencies between the Genesys Info Mart services, but the logic of your particular pipeline might require Genesys Info Mart services to be deployed in a particular order. Depending on the order of deployment, there might be temporary data inconsistencies until all the Genesys Info Mart services are operational. For example, GSP looks for the GCA snapshot when it starts; if GCA has not yet been deployed, GSP will encounter unknown configuration objects and resources until the snapshot becomes available.

There are other private edition services you must deploy before Genesys Info Mart. For detailed information about the recommended order of services deployment, see Order of services deployment.

GDPR support

Not applicable, provided your Kafka retention policies have not been set to more than 30 days. GSP does not store information beyond the ephemeral data used during processing.

GSP Kafka topics

For GSP, topics in Kafka represent various data domains and GSP expects certain topics to be defined.

- If a topic does not exist, GSP will never receive data for that domain.
- If a customized topic has been created but not defined in GSP configuration, data from that domain will be discarded.

Unless Kafka has been configured to auto-create topics, you must manually ensure that all of the Kafka topics GSP requires are created in Kafka configuration.

The following table shows the topic names GSP expects to be available. In this table, an entry in the **Customizable GSP parameter column** indicates support for customizing that topic name.

Topic name	Customizable GSP parameter	Description
GSP consumes data from the following topics:		
designer-sdr	designer	Name of the input topic with Session Detail Record (SDR) data
digital-agentstate	digitalAgentStates	Name of the input topic with digital agent states
digital-itx	digitalItx	Name of the input topic with digital interactions
gca-cfg	cfg	Name of the input topic with configuration data
voice-agentstate		Name of the input topic with voice agent states
voice-callthread		Name of the input topic with voice interactions
voice-outbound		Name of the input topic with outbound (CX Contact) activity associated with either voice or digital interactions
GSP produces data into the following topics:		
gsp-cfg	cfg	Name of the output topic for configuration reporting
gsp-custom	custom	Name of the output topic for custom reporting
gsp-ixn	interactions	Name of the output topic for interactions
gsp-mn	mediaNeutral	Name of the output topic for media-neutral agent states
gsp-outbound	outbound	Name of the output topic for outbound (CX Contact) activity
gsp-sm	agentStates	Name of the output topic for agent states

Configure GSP

Contents

Learn how to configure GIM Stream Processor (GSP).

Related documentation:

-
-
-
-

RSS:

- For private edition

GSP Helm chart overrides

The GSP requires some configuration for deployment that must be made by modifying the GSP's default Helm chart. You do this by creating override entries in the GSP's **values.yaml** file.

Download the GSP Helm charts from your image registry, using the appropriate credentials.

For information about how to download the Helm charts, see Downloading your Genesys Multicloud CX containers. To find the correct Helm chart version for your release, see Helm charts and containers for Genesys Info Mart for the Helm chart version you must download for your release. For general information about Helm chart overrides, see Overriding Helm chart values in the *Genesys Multicloud CX Private Edition Guide*.

At minimum, you must create entries in the **values.yaml** file to specify key system information, as described in the following sections.

> **Important**
>
> Treat your modified **values.yaml** file as source code, which you are responsible to maintain so that your overrides are preserved and available for reuse when you upgrade.

Image registry and pull secret

Image registry

Create an entry in the GSP's **values.yaml** file to specify the location of your image registry. This is

the repository from which Kubernetes pulls images.

The location of the image registry is defined when you set up the environment for the GSP. It is represented in the system as the `docker-registry`. In the GSP Helm chart, the repository is represented as `image: registry`, as shown in the following example. You can optionally set a container version for the image, as in the following example.

```
image: # The repository from which Kubernetes pulls images
  registry:   # The default registry is pureengage-docker-staging.jfrog.io
  tag:  # The container image tag/version
```

Pull secret

When you set up your environment, you define a pull secret for the image registry (`docker-registry`). You must include the pull secret in the GSP's **values.yaml** file in order for Kubernetes to be able to pull from the repository.

```
imagePullSecrets:
  docker-registry: {} # The credentials Kubernetes will use to pull the image from the
registry
```

Note that other services use a different syntax than this to configure the repository pull secret, as follows:

```
imagePullSecrets:
  name: docker-registry
```

Genesys Info Mart, GIM Stream Processor, and GIM Configuration Adaptor helm charts all support advanced templating that allow the helm to create the pull secret automatically; hence the variation in syntax.

Kafka

Kafka secret

If Kafka is configured with authentication, you must configure the Kafka secret so GSP can access Kafka. The Kafka secret is provisioned in the system as `kafka-secrets` when you set up the environment for GSP. Configure the Kafka secret by creating a Helm chart override in the **values.yaml** file.

```
kafka:
  password:  # Credentials for accessing Kafka. This secret is created during deployment.
```

Kafka bootstrap

To allow the Kafka service on GSP to align with the infrastructure Kafka service, make a Helm override entry with the location of the Kafka bootstrap.

```
kafka:
```

```
  bootstrap:  # The Kafka address to align with the infrastructure Kafka
```

Custom Kafka topic names

Some of the Kafka topics used by the GSP support customizing the topic name. If any topic name has been customized, ensure it is represented as a Helm chart override entry, using the `kafka:topic` parameter.

For a list of the Kafka topics that GSP produces and consumes, including which of those support customized naming, see Before you begin GSP deployment.

S3-compatible storage

S3 storage credentials

When you set up the environment for GSP, you provision S3-compatible object storage for GSP to use as a persistent data store. In the **values.yaml** file, record the credentials needed by GSP to access this storage.

```
gsp-s3:  # Credentials for accessing S3-compatible storage
```

Enable S3-compatible storage

When you set up your environment for the GSP, you provision S3-compatible object storage for the GSP's persistent data store. You must enable this storage with override entries in the **values.yaml** file.

By default, GSP is configured to use Azure Blob Storage as the persistent data store. If you have provisioned Azure Blob Storage in your deployment, modify the following entries in the **values.yaml** file:

```
job:
  storage:
    gspPrefix:  # The URI path prefix under which GSP savepoints, checkpoints, and high
availability data will be stored
    gcaSnapshots:  # The URI path under which GCA snapshots will be stored
```

To enable other types of S3-compatible storage, modify the following entries in the **values.yaml** file:

```
azure:
  enabled: false
job:
  storage:
    gspPrefix:  # The bucket where GSP savepoints, checkpoints, and high-availability data
will be stored
    gcaSnapshots:  # The bucket where the GCA snapshots will be stored.
    s3:  # The applicable details defined for the OBC or GCP bucket.
```

Note: The `host` parameter is ignored.

GKE example

```
azure:
  enabled: false
...
job:
  storage:
    host: gspstate{{.Values.short_location}}{{.Values.environment}}.blob.core.windows.net
    #gspPrefix: wasbs://gsp-state@{{ tpl .Values.job.storage.host . }}/{{ .Release.Name }}/
    gspPrefix: "s3p://test-example-bucket-one/{{ .Release.Name
}}/"
    #gcaSnapshots: wasbs://gca@{{ tpl .Values.job.storage.host . }}/
    gcaSnapshots: "s3p://test-example-bucket-one/
gca/"
    checkpoints: '{{ tpl .Values.job.storage.gspPrefix . }}checkpoints'
    savepoints: '{{ tpl .Values.job.storage.gspPrefix . }}savepoints'
    highAvailability: '{{ tpl .Values.job.storage.gspPrefix . }}ha'
    s3:
      endpoint: "https://storage.googleapis.com:443"
      accessKey: ""
      secretKey: ""
      pathStyleAccess: "true"
```

Kubernetes API

GSP uses Apache Flink for stateful stream processing, with communications handled via the Kubernetes API. To use the Kubernetes API, GSP must have the permissions shown in the following example:

Verbs	On Resource	API Group	Comment
get list watch delete	jobs	batch	GSP uses these commands during upgrade and for a pre-upgrade hook to ensure that the previous version of GSP is stopped before upgrading to the new version.
create update patch get list watch delete	configmap	general ("")	GSP uses these commands to: • Support Flink's Kubernetes high availability (HA) services • Record the path to the savepoint (periodically taken by the **take-savepoint** cron job and by the upgrade

Verbs	On Resource	API Group	Comment
			hook)

Configure GSP behavior

You can specify values in the **values.yaml** file to override the default values of configuration options that control GSP behavior and to customize user data and Outbound field mappings.

You can override aspects of the default configuration to modify GSP behavior and customize the way data is stored in the Info Mart database.

You can customize the following:

- Configuration options, to modify data-related aspects of GSP behavior
- User data mappings, to map custom key-value pairs (KVPs), which are attached to event and reporting protocol data, to tables and columns in the Info Mart database
- Outbound record field mappings, to map custom CX Contact record field data to Outbound-related tables and columns in the Info Mart database

Customize configuration option settings

For full information about the options you can configure, including the default and valid values, see GSP configuration options.

To configure options, edit the GSP **values.yaml** file. Under the **cfgOptions** object, specify the option and value in JSON format, noting the following:

- Options are separately configurable by tenant and, where applicable, by media type or even at the level of individual queues (DNs or scripts).
- Where an option can be configured at various levels, you can override a value set at a higher level (for example, for a particular media type in general) to set a different value for a particular lower-level object (for example, for that media type for an individual DN).
- See the note about configuration levels for information about the available configuration levels for certain options.

The entries in the **values.yaml** file are structured as follows:

```
cfgOptions:
    "": |
      standard:

:
      media:
        :

:

:
```

```
        :

:

:
      dn:
        :
          media:
            :

:
      script:
        :
          media:
            :

:
      ...
```

For an example, see example.

Customize user data mapping

Genesys Info Mart uses a wide range of user-data KVPs from a number of upstream services to populate data in the Info Mart database. For full information about user data in Genesys Info Mart, see [[PEC-REP/Current/GIMPEGuide/UserData|]].

As described on [link TBD], you can extend storage of user data in the Info Mart database to include additional user-data KVPs you want to capture as custom user-data facts or dimensions.

To configure user-data mapping, edit the GSP **values.yaml** file. Under the **udeMapping** object , specify the mapping between your custom KVPs and the custom user-data database table(s) and column(s), noting the following:

- The mapping, which is specified in JSON format, is configured separately by tenant.
- In addition to specifying the database table and column in which to store the KVP value, you also specify the *propagation rule* that Genesys Info Mart uses to determine what value to store if more than one value is extracted for the same key in the same interaction. See Propagation rules for more information.
- You can specify a default value to use if the KVP value is null. You must provide a default value for dimensions.
- You map custom user-data dimensions via the IRF_USER_DATA_KEYS table, where you specify the foreign key reference(s) for the user-data dimension table(s).

Genesys Info Mart provides the following sample tables in the Info Mart database schema for storage of custom KVPs:

- USER_DATA_CUST_DIM_1 — for low-cardinality user data to be stored as dimensions
- IRF_USER_DATA_CUST_1 — for high-cardinality user data to be stored as facts

The structure of the entries to specify in the **values.yaml** file is:

```
udeMapping:
  ": |
    IRF_USER_DATA_KEYS:
      columns:
        :
          type: dim
          table:
          attributes:
            :
              kvp:
              rule:
              default:
            ...
    :
      columns:
        :
          type: value
          kvp:
          rule:
          default:
        ...
```

See the example.

Customize Outbound field mapping

Genesys Info Mart stores data about every outbound contact attempt, based on Record Field data it receives from the CX Contact (CXC) service. As described on [[PEC-REP/Current/GIMPEGuide/Outbound|]], some of the mapping between Field data and the Info Mart database tables and columns is predefined, and some is custom.

To customize outbound mapping, edit the GSP values.yaml file. Under the **ocsMapping** object, specify the mapping between your custom record fields and the tables and columns provided in the Info Mart database for custom record field data, namely:

- In the CONTACT_ATTEMPT_FACT table:
  - 10 floating-point numbers: `numeric(14,4)`
  - 20 integers: `integer`
  - 30 strings: `varchar(255)`
- In the RECORD_FIELD_GROUP_1 and RECORD_FIELD_GROUP_2 tables:
  - 10 strings each: `varchar(255)`

The mapping, which is specified in JSON format, is configured separately by tenant. The structure of the entries is:

```
ocsMapping:
  "": |
    CONTACT_ATTEMPT_FACT:
      columns:
        RECORD_FIELD_1:
          field:
          rpcValue:
          conversionValue:
        ...
        RECORD_FIELD_11:
```

```
          field:
          rpcValue:
          conversionValue:
        ...
        RECORD_FIELD_31:
          field:
          rpcValue:
          conversionValue:
        ...
    RECORD_FIELD_GROUP_1:
      columns:
        RECORD_FIELD_1_STRING_1:
          field:
          rpcValue:
          conversionValue:
        ...
    RECORD_FIELD_GROUP_2:
      columns:
        RECORD_FIELD_2_STRING_1:
          field:
          rpcValue:
          conversionValue:
        ...
```

See the example.

Example

```
cfgOptions:
    "eb0c9f3a-5dca-498f-98d5-7610f5fd1015": |
      standard:
        completed-queues: iWD_Completed,iWD_Processed_ext
        populate-workbin-as-hold: false
      media:
        voice:
          q-answer-threshold: 30
      media:
        email:
          q-short-abandoned-threshold: 20
      dn:
        eb0c9f3a-5dca-498f-98d5-7610f5fd1015_MM_VQ:
          media:
            email:
              q-answer-threshold: 90
              q-short-abandoned-threshold: 40
      script:
        eb0c9f3a-5dca-498f-98d5-7610f5fd1015_ixn_queue:
          media:
            chat:
              q-answer-threshold: 20
      script:
        eb0c9f3a-5dca-498f-98d5-7610f5fd1015_dev.IQ:
          standard:
            populate-ixnqueue-facts: false

udeMapping:
  "eb0c9f3a-5dca-498f-98d5-7610f5fd1015": |
    IRF_USER_DATA_KEYS:
      columns:
        CUSTOM_KEY_1:
          type: dim
```

```
        table: USER_DATA_CUST_DIM_1
        attributes:
          DIM_ATTRIBUTE_1:
            kvp: BusinessLine
            rule: CALL
            default: retail
          DIM_ATTRIBUTE_2:
            kvp: RULE_PARTY
            rule: PARTY
            default: none
          DIM_ATTRIBUTE_3:
            kvp: RULE_IRF
            rule: IRF
            default: none
          DIM_ATTRIBUTE_4:
            kvp: RULE_IRF_INITIAL
            rule: IRF_INITIAL
            default: none
          DIM_ATTRIBUTE_5:
            kvp: RULE_IRF_FIRST_UPDATE
            rule: IRF_FIRST_UPDATE
            default: none
  IRF_USER_DATA_CUST_1:
    columns:
      CUSTOM_DATA_1:
        type: value
        kvp: RULE_CALL
        rule: CALL
        default: none
      CUSTOM_DATA_2:
        type: value
        kvp: RULE_PARTY
        rule: PARTY
        default: none
      CUSTOM_DATA_3:
        type: value
        kvp: RULE_IRF
        rule: IRF
      CUSTOM_DATA_4:
        type: value
        kvp: RULE_IRF_FIRST_UPDATE
        rule: IRF_FIRST_UPDATE
      CUSTOM_DATA_5:
        type: value
        kvp: RULE_IRF_INITIAL
        rule: IRF_INITIAL
      CUSTOM_DATA_6:
        type: value
        kvp: RULE_IRF_ROUTE
        rule: IRF_ROUTE
  IRF_USER_DATA_GEN_1:
    columns:
      CASE_ID:
        type: value
        kvp: CaseID
        rule: Call
        default: none
      CUSTOMER_ID:
        type: value
        kvp: GIM_GRP
        rule: Call
        default: none
```

```
ocsMapping:
  "eb0c9f3a-5dca-498f-98d5-7610f5fd1015": |
    CONTACT_ATTEMPT_FACT:
      columns:
        RECORD_FIELD_1:
          field: GenRecordField1
          rpcValue:
          conversionValue:
        RECORD_FIELD_2:
          field: GenRecordField2
          rpcValue:
          conversionValue:
        ...
        RECORD_FIELD_60:
          field: GenRecordField60
          rpcValue:
          conversionValue:
    RECORD_FIELD_GROUP_1:
      columns:
        RECORD_FIELD_1_STRING_1:
          field: GenRecordField1String1
          rpcValue:
          conversionValue:
        RECORD_FIELD_1_STRING_2:
          field: GenRecordField1String2
          rpcValue:
          conversionValue:
        RECORD_FIELD_1_STRING_3:
        ...
    RECORD_FIELD_GROUP_2:
      columns:
        RECORD_FIELD_2_STRING_1:
          field: GenRecordField2String1
          rpcValue:
          conversionValue:
        RECORD_FIELD_2_STRING_2:
        ...
```

GSP configuration options

Contents

Options you can use to control GSP behavior.

Related documentation:

-
-
-
-

RSS:

- For private edition

GSP supports the configuration options described on this page. See Configure GSP behavior for information about how to change the option values in the Helm chart.

GSP behavior affects data in the Info Mart database dimensional model. For full information about the Info Mart database tables and columns referenced in the descriptions on this page, see the *Genesys Info Mart Historical Database Reference*.

- adjust-vq-time-by-strategy-time
- canceled-queues
- completed-queues
- interaction-type-ignore-list
- max-ixn-duration-days
- max-msfs-per-irf
- populate-ixnqueue-facts
- populate-workbin-as-hold
- populate-workbin-facts
- q-answer-threshold
- q-short-abandoned-threshold
- short-abandoned-threshold
- show-non-queue-mediation
- stop-ixn-queues

Important

When you specify an override value in the **values.yaml** file, you indicate the configuration level for that option value. The configuration level specifies the scope for which an option value is in effect. For example, for certain ***-threshold** options, you can specify different thresholds for the various media types, and you can further refine behavior by specifying different thresholds for individual queues. Only some options support configuration at different levels.

All options are configured by tenant. Below the tenant level, the available configuration levels are:

- standard — Applies across media types.
- media — Applies to the specified media type, such as voice or email.

- dn: media — Applies to the specified Virtual Queue DN. An option value specified on the DN level overrides a value specified for that option at the media level.
- script: media *or* script: standard — Applies to the specified Script object representing an Interaction Queue or Interaction Workbin for digital interactions. An option value specified on the Script level overrides a value specified for that option at the media or standard levels.

adjust-vq-time-by-strategy-time

Default value: false
Valid values: true, false
Configuration level: standard

Specifies whether GSP adjusts the mediation duration in mediation segment facts (MSFs) for virtual queues to include time spent in strategies but not in associated virtual queues — for example, if an interaction spends 3 minutes in a strategy and is in a virtual queue for 2 of those minutes, whether the MSF for the virtual queue reports the duration as 3 minutes or 2 minutes.

- `true` — GSP includes strategy time that is outside the virtual queue in the mediation duration in MSFs for virtual queues.
- `false` — GSP never includes strategy time that is outside the virtual queue in the mediation duration. This setting means that there might be gaps between the end time of the MSF for a virtual queue that is used by a strategy and the interaction resource fact (IRF) that follows the strategy's routing, or between the start time of the MSF for a virtual queue and the end time of a previous MSF (for example, the MSF for an Interaction Workbin).

In digital scenarios in which an interaction is bounced between a mediation resource (for example, an Interaction Queue or a Workbin) and a strategy as the strategy repeatedly retries busy agents, this option comes into effect only for the virtual queue that is associated with the last strategy party, before the interaction is routed successfully or else terminated. Genesys Info Mart does not report on virtual-queue activity that overlaps the repeated interim mediations.

>> Back to list

canceled-queues

Default value: iWD_Canceled
Valid values: A comma-separated list of queue names
Configuration level: standard

In digital deployments that use archive queues in their business processes, specifies the Interaction Queues that are used as archives for canceled interactions. When an interaction is placed into one of these queues, GSP considers the interaction to be terminated. The transformation job assigns the

technical result/reason combination of COMPLETED/CANCELED in the IRF of the handling resource that placed the interaction in the queue, and GSP excludes the interaction from further processing.

The option parallels the **completed-queues** configuration option available in Interaction Server. The default value of the GSP option matches the archive queue for canceled interactions in the default business process for the Genesys intelligent Workload Distribution (iWD) solution.

>> Back to list

completed-queues

Default value: iWD_Completed
Valid values: A comma-separated list of queue names
Configuration level: standard

In digital deployments that use archive queues in their business processes, specifies the Interaction Queues that are used as archives for completed interactions. When an interaction is placed into one of these queues, GSP considers the interaction to be terminated. The transformation job assigns the technical result/reason combination of COMPLETED/ARCHIVED in the IRF of the handling resource that placed the interaction in the queue, and GSP excludes the interaction from further processing.

The option parallels the **completed-queues** configuration option available in Interaction Server. The default value of the GSP option matches the archive queue for completed interactions in the default business process for the Genesys iWD solution.

>> Back to list

interaction-type-ignore-list

Default value: InboundReport, InboundDisposition
Valid values: A comma-separated list of interaction types
Configuration level: standard

In digital deployments, specifies the interaction types or subtypes you want to exclude from reporting. GSP does not transform any interactions with the interaction type(s) or subtype(s) you specify. No records are generated in fact tables for interactions of this type.

If you disable a subtype, the parent interactions of that subtype as well as any child interactions of those parent interactions are disabled, even if the child interactions themselves are of a different subtype, one that is configured to be transformed.

>> Back to list

max-ixn-duration-days

Default value: 30
Valid values: Any positive integer
Configuration level: standard

Specifies the maximum duration, in days, of interactions in the deployed environment.

For long-living digital interactions, specifies the maximum number of days to process active interactions, after which the interactions become eligible for artificial termination. Consider using a value that is consistent with the value of the option **days-to-keep-active-facts**.

Artificial termination of long-living interactions

This functionality is intended primarily to prevent problems with aggregation — for example, when updates to long-living digital interactions trigger unnecessary re-aggregation, which results in overflow errors.

If an interaction is still active when the **max-ixn-duration-days** period expires, the transformation job terminates the interaction artificially in fact tables. Any activity that is related to this interaction that occurs after termination is discarded.

>> Back to list

max-msfs-per-irf

Default value: 50
Valid values: 10-10000
Configuration level: standard

Specifies the maximum number of MSF records associated with a given digital interaction that are represented in the Info Mart database. When the number of MSF records associated with a single IRF record exceeds the limit, the transformation job processes only the first n mediation DNs and the last n mediation DNs in the interaction, where n = max-msfs-per-irf/2. In this way, ETL performance avoids being degraded by huge numbers of MSF records for unsuccessful routing attempts for "stuck strategy" scenarios.

The first n mediation records for a given interaction are processed by the transformation job and populated in the MSF table as they occur. The last n records are postponed until an associated IRF record has been created.

The option does not affect the mediation durations that are reported in the IRF (QUEUE_DURATION, ROUTING_POINT_DURATION, MEDIATION_DURATION, and PREVIOUS_MEDIATION_DURATION); all these metrics correctly report the full overall mediation time.

Log message number 55-20120 is generated when the max-msfs-per-irf limit is exceeded, triggering the behavior to abbreviate the representation of unsuccessful routing attempts in the Info Mart database. You can set an alarm on the log message, to prompt you to investigate strategies that might be inappropriate for your deployment.

>> Back to list

populate-ixnqueue-facts

Default value: true
Valid values: true, false
Configuration level: standard, script: standard

Enables or disables the population of digital Interaction Queue activity to the MSF table. GSP uses the value configured at the `standard` level for all configured Interaction Queues, but you can override that value for specific Interaction Queues by configuring a different value for those Interaction Queues at the `script: standard` level.

- `true` — The placement of an interaction in an Interaction Queue is represented in the MSF table.
- `false` — The placement of an interaction in an Interaction Queue is not represented in the MSF table.

GSP always creates an MSF record for the first Interaction Queue that an inbound interaction enters, regardless of the setting of this option.

Strategy time is not included in the mediation duration, with the following exceptions:

- Scenarios in which an interaction is bounced between an Interaction Queue and a strategy while an available agent is being identified
- When the **show-non-queue-mediation** option is set to `true`

> ### Tip
> Ideally, set **populate-ixnqueue-facts** to `true` only for Interaction Queues for which mediation reporting is meaningful, so that the MSF table does not become cluttered with unnecessary information. Some Interaction Queues are not useful for reporting, such as the Interaction Queues that are associated with Interaction Workbin objects only through a configured Interaction Queue View.

>> Back to list

populate-workbin-as-hold

Default value: true
Valid values: true, false
Configuration level: standard

Specifies whether the time that an interaction is in an Interaction Workbin is considered to be hold time or mediation.

- `true` — Workbin time is considered to be hold if the handling resource places the interaction into its own personal workbin. The hold ends when any resource takes the interaction out of the workbin. Various hold metrics are reported in the IRF record that is associated with that handling resource.
- `false` — Workbin time is considered to be mediation. Whether or not the workbin activity is represented by an MSF record depends on the value of the **populate-workbin-facts** option.

For time in an Interaction Workbin to be considered hold time, the Agent or Place resource must place the interaction into its own personal workbin. For example, if Agent1 places an email interaction into its personal workbin, the workbin time is considered to be hold; however, if another resource, such as another agent or a strategy, places the interaction into the workbin for Agent1 to handle, the workbin time is not considered to be hold.

If a handling resource places an interaction into more than one of its own personal workbins, all of the personal workbin time is considered to be hold, and there is no distinction between the hold time in the various workbins. For example, if Agent1 places an email interaction into a Drafts workbin, then pulls the interaction to continue handling it and subsequently places it into a FollowUp workbin, the hold duration that is reported in the IRF record for Agent1's association with the interaction combines the time that is spent in both workbins.

The following table summarizes the results of different permutations of the **populate-workbin-as-hold** and **populate-workbin-facts** options, in combination with the specifics of the interaction flow. If the values of the **populate-workbin-as-hold** and **populate-workbin-facts** options are both set to `false`, the workbin activity is not represented in the dimensional model.

Matrix of workbin reporting results

<table>
<tr><th>Scenario</th><th colspan="3">Reporting result for workbin activity</th></tr>
<tr><th>populate-workbin-as-hold=false

populate-workbin-facts=true</th><th>populate-workbin-as-hold=true

populate-workbin-facts=true</th><th colspan="2">populate-workbin-as-hold=true

populate-workbin-facts=false</th></tr>
<tr><td>An Agent/Place resource places the interaction into its personal workbin.</td><td rowspan="3">MSF record.*</td><td colspan="2">Hold metrics in the IRF record for the association of that Agent/Place with the interaction.

This is the default behavior.</td></tr>
<tr><td>Another resource places the interaction into the personal workbin of an Agent/Place.</td><td rowspan="2">MSF record.*

This is the default behavior.</td><td rowspan="2">Not represented in the dimensional model.</td></tr>
<tr><td>Any resource places the interaction into an AgentGroup or PlaceGroup workbin.</td></tr>
<tr><td colspan="4">*The MSF record includes a <code>WORKBIN_KEY</code> value that identifies the workbin instance that is associated with the mediation.</td></tr>
</table>

>> Back to list

populate-workbin-facts

Default value: true
Valid values: true, false
Configuration level: standard, script: standard

Enables or disables the population of digital Interaction Workbin activity to the MSF table. For workbins that are associated with handling resources of type Agent or Place, this option comes into effect only if GSP has not been configured to consider workbin time as hold (`populate-workbin-as-hold=false`). For the circumstances under which GSP considers workbin time as hold, see the description of the **populate-workbin-as-hold** option.

GSP uses the value configured at the `standard` level for all configured Interaction Workbins, but you can override that value for specific Interaction Workbins by configuring a different value for those Interaction Workbins at the `script: standard` level.

- `true` — Provided that GSP does not consider the workbin time as hold, the placement of an interaction in an Interaction Workbin is represented in the MSF table. The MSF record references a WORKBIN dimension, which identifies the type of resource that is associated with the workbin and the specific resource that is associated with the mediation. Strategy time is not included in the mediation duration, except for scenarios in which an interaction is bounced between an Interaction Workbin and a strategy as the strategy repeatedly retries busy agents. For more information, see the description of the MEDIATION_DURATION field in Populating mediation segments.
- `false` — The placement of an interaction in an Interaction Workbin is not represented in the MSF table.

>> Back to list

q-answer-threshold

Default value: 60
Valid values: 1-600000 for digital, 1-10000 for voice.
Configuration level: media, dn: media, script: media

Specifies the default duration, in seconds, that is used on all configured queues as a target time to accept an interaction that entered a queue.

You can configure a different threshold for each media type.

GSP uses the value configured at the `media` level for all configured queues, but you can override the value for interactions of that media type that come through particular virtual queues, digital interaction queues, or workbins by configuring a different value:

- For specific Virtual Queue DNs at the `dn: media` level
- For specific digital Interaction Queues or Interaction Workbins at the `script: media` level

>> Back to list

q-short-abandoned-threshold

Default value: 10
Valid values: 1-1000
Configuration level: media, dn: media, script: media

Specifies the maximum duration of mediation, in seconds, that is used on all configured queues to indicate that an interaction that was abandoned while in a queue should be considered a “short” abandon. GSP uses this value to determine the state of SHORT_ABANDONED_FLAG in the MSF row for an interaction abandoned in a queue.

You can configure a different threshold for each media type.

GSP uses the value configured at the `media` level for all configured queues, but you can override the value for interactions of that media type that come through particular virtual queues, digital interaction queues, or workbins by configuring a different value:

- For specific Virtual Queue DNs at the `dn: media` level
- For specific digital Interaction Queues or Interaction Workbins at the `script: media` level

>> Back to list

short-abandoned-threshold

Default value: 10
Valid values: 0-100
Configuration level: media

Specifies the minimum duration, in seconds, of an abandoned interaction in order for it to be considered truly abandoned. GSP uses this value to determine the state of SHORT_ABANDONED_FLAG in the IRF row.

You can configure a different threshold for each media type.

>> Back to list

show-non-queue-mediation

Default value: false
Valid values: true, false
Configuration level: standard

Controls whether all mediation time for digital interactions, even mediation time that does not occur within a queue, is represented by an MSF.

- `true` — Provided that there is an MSF for the first Interaction Queue in a mediation (which is true for mediation time of unhandled interactions), all mediation time for a digital interaction that occurs after

the first Interaction Queue, even mediation time that does not occur within a queue, is represented by one or more MSFs. Additional, non-queue MSFs are created `for multimedia interactions` to represent mediation time that occurs outside an Interaction Queue MSF — for example, mediation time that occurs after an MSF for an Interaction Queue if a routing strategy is attempting to find a routing target without the use of a virtual queue.

- `false` — MSFs for digital interactions are focused only on the portion of the mediation time that occurs in a queue, whether it is an Interaction Queue or a virtual queue. There may be gaps in time between MSFs, because the interaction may not be in a queue, or may not be in a queue that is represented by an MSF.

For digital interactions, mediation that occurs in a virtual queue is always represented in Genesys Info Mart by an MSF. However, depending on the setting for **populate-ixnqueue-facts**, mediation that occurs in an Interaction Queue might not be represented by an MSF; in fact, often it is not. When **show-non-queue-mediation** is set to `true`, the additional MSFs that are created occur between MSFs for Interaction Queues (in other words, when an interaction moves during mediation from one Interaction Queue that is represented by an MSF to another that is represented by an MSF), or between the MSF for an Interaction Queue and a routing target (agent). The additional, non-queue MSFs may overlap with MSFs for virtual queues, since an interaction may also be in a virtual queue for some (or all) of the mediation time that occurs outside of an Interaction Queue MSF. Furthermore, the additional MSFs may include time that the interaction spent in Interaction Queues that are not represented by an MSF.

Example

Consider the scenario in which a digital interaction enters the contact center at time t0 and, after mediation involving various queues and routing strategies, is routed for handling at time t4. Following first handling, there is additional mediation before the interaction is routed for further handling.
t0–t1: InteractionQueue1
t1–t2: Strategy1
t2–t3: InteractionQueue2 (not the same queue as InteractionQueue1)
t3–t4: Strategy2
t4–t5: Agent1
t5–t6: InteractionQueue3
t6–t7: Strategy3
t7–t8: InteractionQueue4
t8–t9: Strategy4
t9–t10: Agent2

The following table summarizes the mediation reporting results, depending on configuration option settings.

populate-ixnqueue-facts=false, **show-non-queue-mediation=false**	**populate-ixnqueue-facts=true,** **show-non-queue-mediation=false**	**populate-ixnqueue-facts=false,** **show-non-queue-mediation=true**	**populate-ixnqueue-facts=true,** **show-non-queue-mediation=true**
Without virtual queues			
• MSF1 (InteractionQueue1)* • IRF1 (Agent1)	• MSF1 (InteractionQueue1) • MSF2	• MSF1 (InteractionQueue1) • MSF2 (Strategy1)**	• MSF1 (InteractionQueue1) • MSF2 (Strategy1)

populate-ixnqueue-facts=false, **show-non-queue-mediation=false**	**populate-ixnqueue-facts=true,** **show-non-queue-mediation=false**	**populate-ixnqueue-facts=false,** **show-non-queue-mediation=true**	**populate-ixnqueue-facts=true,** **show-non-queue-mediation=true**
• IRF2 (Agent2) *If the second InteractionQueue in the scenario is actually the same as InteractionQueue1, then MSF1 would cover t0–t3, and there would be a gap of t3–t4.	(InteractionQueue2) • IRF1 (Agent1) • MSF3 (InteractionQueue3) • MSF4 (InteractionQueue4) • IRF2 (Agent2)	• IRF1 (Agent1) • IRF2 (Agent2) **Spans t1–t4, including time for InteractionQueue2 and Strategy2.	• MSF3 (InteractionQueue2) • MSF4 (Strategy2) • IRF1 (Agent1) • MSF5 (InteractionQueue3) • MSF6 (Strategy3) • MSF7 (InteractionQueue4) • MSF8 (Strategy4) • IRF2 (Agent2)
Gaps: t1–t4, t5–t9	**Gaps:** t1–t2, t3–t4, t6–t7, t8–t9	**Gaps:** None up to first handling; t5–t9	**Gaps:** None
If the strategies use virtual queues, there are also separate MSFs for the virtual queues, which might eliminate gaps, even when **show-non-queue-mediation**=`false`, and which overlap with the MSF for the Strategy party when **show-non-queue-mediation**=`true`.			

> **Important**
>
> There are other scenarios, not controlled by this option, where non-queue time might be incorporated into an MSF for a queue, such as when a digital interaction bounces back and forth between the same queue and a strategy. In such cases, the repeated bounce-backs are collapsed into a single MSF, which might include non-queue time when the routing strategy was attempting to find a routing target before returning the interaction to the same queue. For more information, see the description of the MEDIATION_DURATION field in Populating mediation segments.

>> Back to list

stop-ixn-queues

Default value: No default value
Valid values: A comma-separated list of queue names
Configuration level: standard

In digital deployments that use stop-interaction queues in their business processes, this option

specifies the Interaction Queues that are used to handle stopping an interaction (for example, Twitter_StopIxn). When an interaction is placed into one of these queues, GSP considers the interaction to be terminated:

- The transformation job assigns the technical result/reason combination of COMPLETED/UNSPECIFIED in the IRF of the handling resource that placed the interaction in the queue, and GSP excludes the interaction from further processing.
- The agent who placed the interaction in the queue is represented as the party that stopped the interaction, and the strategy that actually stops the interaction and performs any associated post-processing is not represented in Genesys Info Mart reporting.

If you do not specify a value for this option, GSP handles these situations as transfers to a queue.

>> Back to list

Deploy GIM Stream Processor

Contents

Learn how to deploy GIM Stream Processor (GSP) into a private edition environment.

Related documentation:

-
-
-
-

RSS:

- For private edition

Assumptions

- The instructions on this page assume you are deploying the service in a service-specific namespace, named in accordance with the requirements on Creating namespaces. If you are using a single namespace for all private edition services, replace the namespace element in the commands on this page with the name of your single namespace or project.
- Similarly, the configuration and environment setup instructions assume you need to create namespace-specific (in other words, service-specific) secrets. If you are using a single namespace for all private edition services, you might not need to create separate secrets for each service, depending on your credentials management requirements. However, if you do create service-specific secrets in a single namespace, be sure to avoid naming conflicts.

> **Important**
>
> Make sure to review Before you begin GSP deployment for the full list of prerequisites required to deploy GSP, including provisioning S3-compatible storage (see Before you begin GSP deployment).

Set up your environment

To prepare your environment for the deployment, complete the steps in this section. Use the instructions for your environment:

- GKE
- AKS

GKE environment setup

1. Ensure that the gcloud CLI and required Helm version are installed on the host where you will run the deployment.
2. Using the CLI, log in to the GKE cluster on the host where you will run the deployment.

   ```
   gcloud container clusters get-credentials
   ```

3. If the cluster administrator has not already done so, create a new namespace `gsp` for GSP.
   - Create a JSON file that specifies the namespace metadata. As an example, the file **create-gsp-namespace.json.** has the following JSON

     ```
     {
        "apiVersion": "v1",
        "kind": "Namespace",
        "metadata": {
          "name": "gsp",
          "labels": {
            "name": "gsp"
          }
        }
     }
     ```

   - Create the namespace. The following example applies the **create-gsp-namespace.json** file to create the `gsp` namespace.

     ```
     kubectl apply -f create-gsp-namespace.json
     ```

   - Confirm that the namespace was successfully created.

     ```
     kubectl describe namespace gsp
     ```

4. Create the pull secret for the image registry.
   - This step defines a secret so that Kubernetes can authenticate your image repository and pull artifacts from it. The repository is represented as `docker-registry` in the system. For information about downloading artifacts from the repository, see Downloading your Genesys Multicloud CX containers.
   - When you configure the GSP, you will reference the registry pull secret as a Helm chart override; see GSP Helm chart overrides.

     ```
     kubectl create secret docker-registry  --docker-server= --docker-username= --docker-
     password= --docker-email= -n gsp
     ```

5. Create the Kafka secret.
   - Kafka requires a separate access credential.
   - This credential is represented as `kafka-secrets` in the system, as in the following syntax.

     ```
     kubectl create secret generic kafka-secrets --from-literal=kafka-
     secrets={\"bootstrap\":\"\", \"username\":\"gsp\",\"password\":\"\"} -n gsp
     ```

     For example:

     ```
     kubectl create secret generic kafka-secrets --from-literal=kafka-
     secrets={\"bootstrap\":\"infra-kafka-cp-
     kafka.infra.svc.cluster.local:9092\",\"username\":\"gsp\",\"password\":\"kafka-
     password\"} -n gsp
     ```

Note: Kafka can be deployed without authentication, as in the following example.

```
kubectl create secret generic kafka-secrets --from-literal=kafka-
 secrets={\"bootstrap\":\"\"} -n gsp
```

AKS environment setup

1. Ensure that the Azure CLI and required Helm version are installed on the host where you will run the deployment.
2. Using the CLI, log in to the cluster on the host where you will run the deployment.

   ```
   az aks get-credentials --resource-group  --name  --admin
   ```

3. If the cluster administrator has not already done so, create a new namespace `gsp` for GSP.
   - Create a JSON file that specifies the namespace metadata. As an example, the file **create-gsp-namespace.json** has the JSON shown below.

     ```
     {
        "apiVersion": "v1",
        "kind": "Namespace",
        "metadata":
          "name": "gsp",
          "labels": {
            "name": "gsp"
          }
        }
      }
     ```

   - Create the namespace. The following example applies the **create-gsp-namespace.json** file to create the `gsp` namespace.

     ```
     kubectl apply -f create-gsp-namespace.json
     ```

   - Confirm that the namespace was successfully created.

     ```
     kubectl describe namespace gsp
     ```

4. Create the pull secret for the image registry.
   - This step defines a secret so that Kubernetes can authenticate your image repository and pull artifacts from it. The repository is represented as `docker-registry` in the system. For information about downloading artifacts from the repository, see Downloading your Genesys Multicloud CX containers.
   - When you configure the GSP, you will reference the registry pull secret as a Helm chart override; see GSP Helm chart overrides.

     ```
     kubectl create secret docker-registry  --docker-server= --docker-username= --docker-
      password= --docker-email= -n gsp
     ```

5. Create the Kafka secret.
   - Kafka requires a separate access credential.
   - This credential is represented as `kafka-secrets` in the system, as in the following syntax.

     ```
     kubectl create secret generic kafka-secrets --from-literal=kafka-
      secrets={\"bootstrap\":\"\", \"username\":\"gsp\",\"password\":\"\"} -n gsp
     ```

For example:

```
kubectl create secret generic kafka-secrets --from-literal=kafka-
 secrets={\"bootstrap\":\"infra-kafka-cp-
 kafka.infra.svc.cluster.local:9092\",\"username\":\"gsp\",\"password\":\"kafka-
 password\"} -n gsp
```

Note: Kafka can be deployed without authentication, as in the following example.

```
kubectl create secret generic kafka-secrets --from-literal=kafka-
 secrets={\"bootstrap\":\"\"} -n gsp
```

Deploy the GSP

After the environment has been set up, deploying the GSP is a matter of executing the following command:

```
helm install gsp  -f  -n gsp
```

Validate the deployment

You can consider GSP deployment successful when the pod is running and in Ready state. Genesys Info Mart does not report the Ready state for pods until internal health checks are satisfied and the pods are operational. You can use standard `kubectl` commands like `list` and `get` to verify the successful deployment and readiness status of the Kubernetes objects.

However, from a functional point of view, you cannot validate GSP deployment unless GCA and GIM have been deployed as well. Do not expect consistent data until all three Genesys Info Mart services are up and running. When all three services have been deployed:

1. Make a few test calls employing different routing strategies under different scenarios, and verify that all the calls are correctly captured in the Info Mart database. For example:
   - The calls appear in the interaction-related tables.
   - The calls have been correctly assigned to agents and queues.
2. Review the logs to verify no errors.
3. Monitor the operations dashboard to verify that the services report their status as Ready, and pods are not continually restarting.

Before you begin GIM deployment

Contents

Find out what to do before deploying GIM.

Related documentation:

-
-
-
-

RSS:

- For private edition

Limitations and assumptions

Instructions are provided for a single-tenant deployment.

Download the Helm charts

To configure and deploy Genesys Info Mart, you must obtain the Helm charts included with the Info Mart release. These Helm charts provision Info Mart plus any Kubernetes infrastructure Info Mart requires to run.

Genesys Info Mart and GIM monitoring are the only services that run in the Info Mart container.

For the correct Helm chart version for your release, see Helm charts and containers for Genesys Info Mart. For information on downloading from the image repository, see Downloading your Genesys Multicloud CX containers.

Third-party prerequisites

For information about setting up your Genesys Multicloud CX private edition platform, see Software requirements.

The following table lists the third-party prerequisites for GIM.

Third-party services

Name	Version	Purpose	Notes
PostgreSQL	11.x	Relational database.	You must create the Info Mart database (see

Name	Version	Purpose	Notes
			Before you begin GIM deployment).
Kafka	2.x	Message bus.	The Kafka topics GIM will consume must exist in the Kafka configuration. GIM consumes the topics GSP produces. For more information, see GSP Kafka topics.
Object storage		Persistent or shared data storage, such as Amazon S3, Azure Blob Storage, or Google Cloud Storage.	(Optional) For the Data Export feature, GIM uses object storage to store the exported data. Alternatively, you can elect to store the exported data in a directory on a local machine.
A container image registry and Helm chart repository		Used for downloading Genesys containers and Helm charts into the customer's repository to support a CI/CD pipeline. You can use any Docker OCI compliant registry.	
Command Line Interface		The command line interface tools to log in and work with the Kubernetes clusters.	

Storage requirements

GIM uses PostgreSQL for the Info Mart database and, optionally, uses object storage to store exported Info Mart data.

PostgreSQL — the Info Mart database

The Info Mart database stores data about agent and interaction activity, Outbound Contact campaigns, and usage information about other services in your contact center. A subset of tables and views created, maintained, and populated by Reporting and Analytics Aggregates (RAA) provides the aggregated data on which Genesys CX Insights (GCXI) reports are based.

A sizing calculator for Genesys Multicloud CX private edition is under development. In the meantime, the interactive tool available for on-premises deployments might help you estimate the size of your Info Mart database, see *Genesys Info Mart 8.5 Database Size Estimator*.

Genesys recommends a minimum 3 IOPS per GB.

For information about creating the Info Mart database, see Before you begin GIM deployment

Create the Info Mart database

Use any database management tool to create the Info Mart ETL database and user.

1. Create the database.
2. Create a user for all of the Genesys Info Mart services to use. Grant that user full permissions for the database.
   This user's account is used by Info Mart jobs to access the Info Mart database schema.

   The name of the Info Mart schema name is `public`.

   > **Important**
   > Make a note of the database and user details. You need this information when you configure GIM and GCA Helm chart override values.

Object storage — Data Export packages

The GIM Data Export feature enables you to export data from the GIM database so it is available for other purposes. Unless you elect to store your exported data in a local directory, GIM data is exported to an object store. GIM supports export to either Azure Blob Storage or the S3-compatible storage provided by Google Cloud Platform (GCP).

If you want to use S3-compatible storage, follow the Before you begin GSP deployment instructions for GSP to create the S3-compatible storage for GIM.

> **Important**
> GSP and GCA use object storage to store data during processing. For safety and security reasons, Genesys strongly recommends that you use a dedicated object storage account for the GIM persistent storage, and do not share the storage account created for GSP and GCA. GSP and GCA can share an account, and this is the expected deployment.

If you are not using obect storage, you can configure GIM to store exported data in a local directory. In this case, you do not need to create the object storage.

Network requirements

No special network requirements.

Browser requirements

Not applicable

Genesys dependencies

- You must have your Tenant ID information available.
- There are no strict dependencies among the Genesys Info Mart services, but the logic of your particular pipeline might require them to be deployed in a particular order. Depending on the order of deployment, there might be temporary data inconsistencies until all the Genesys Info Mart services are operational. For example, GCA might try to access the Info Mart database to synchronize configuration data but if GIM has not yet been deployed, the Info Mart database will be empty.

For detailed information about the correct order of services deployment, see Order of services deployment.

GDPR support

GIM provides full support for you to comply with Right of Access ("export") or Right of Erasure ("forget") requests from consumers and employees with respect to personally identifiable information (PII) in the Info Mart database.

Genesys Info Mart is designed to comply with General Data Protection Regulation (GDPR) policies. Support for GDPR includes the following:

- The way that Genesys Info Mart processes customer files complies with Right of Access ("export") and Right of Erasure ("forget") requirements.
- Genesys Infor Mart supports configuring data retention policies.
- GDPR processing is fully audited.

For more information about how Genesys Info Mart implements support for GDPR requests, see [[PEC-REP/Current/GIMPEGuide/GDPR|]].

For details about the Info Mart database tables and columns that potentially contain PII, see the description of the CTL_GDPR_HISTORY table in the Genesys Info Mart on-premises documentation.

Configure GIM

Contents

Learn how to configure GIM.

Related documentation:

-
-
-
-

RSS:

- For private edition

GIM Helm chart overrides

Genesys Info Mart requires some configuration for deployment that you can make only by modifying the Helm chart, which you do by creating override entries in the GIM **values.yaml** file.

Download the **gim** and **gim-monitoring** Helm charts from your image repository, using the appropriate credentials.

To learn how to download the Helm charts, see Downloading your Genesys Multicloud CX containers. To find the correct Helm chart version for your release, see Helm charts and containers for Genesys Info Mart. For general information about Helm chart overrides, see Overriding Helm chart values in the *Genesys Multicloud CX Private Edition Guide*.

At minimum, you must create entries in the **values.yaml** file to specify system information, as described in the following sections.

> **Important**
>
> Treat your modified **values.yaml** file as source code, which you are responsible to maintain so that your overrides are preserved and available for reuse when you upgrade.

Image repository and pull secret

Image registry

To specify the location of the image registry, create an entry in the GIM **values.yaml** file. This is the

repository from which Kubernetes will pull images.

The location of the image registry is defined when you set up the environment for the GIM, and is represented in the system as the `docker-registry`. In the GIM Helm chart, the repository is represented as `image: registry`, as shown in the following example. You can optionally set a container version for the image.

```
image: # The repository from which Kubernetes will pull images
  registry:   # The default registry is pureengage-docker-staging.jfrog.io
  tag:     # the container image tag/version
```

Pull secret

When you set up your environment, you define a pull secret for image registry (`docker-registry`). You must include the pull secret in the GIM **values.yaml** file for Kubernetes to be able to pull from the repository.

```
imagePullSecrets:
  docker-registry: {} # The credentials Kubernetes will use to pull the image from the
registry
```

Other services use a different syntax to configure the repository pull secret, as follows:

```
imagePullSecrets:
  name: docker-registry
```

Genesys Info Mart, GIM Stream Processor, and GIM Configuration Adapter helm charts all support advanced templating that allow the helm to create the pull secret automatically; hence the variation in syntax.

Kafka

Kafka secret

If Kafka is configured with authentication, you must configure the Kafka secret so the GIM service can access Kafka. The Kafka secret is provisioned in the system as `kafka-secrets` when you set up the environment for Info Mart. Configure the Kafka secret by creating a Helm chart override in the **values.yaml** file.

```
kafka:
  password:  # Credentials for accessing Kafka. This secret is created during deployment.
```

Kafka bootstrap

To allow the Kafka service on Info Mart to align with the infrastructure Kafka service, make a Helm override entry with the location of the Kafka bootstrap.

```
kafka:
  bootstrap:  # the Kafka address to align with the infrastructure Kafka
```

Custom Kafka topic names

Some of the Kafka topics used by the GSP support customizing the topic name. If any topic name has been customized, ensure it is represented as a GIM Helm chart override entry, using the `kafka:topic` parameter.

For a list of the Kafka topics that GSP produces and consumes, including which of those support customized naming, see Before you begin GSP deployment.

Data export and S3-compatible storage

If the Genesys Info Mart Data Export feature is part of your deployment, you can export your data to S3-compatible object storage.

You provision the storage when you set up the environment for Genesys Info Mart, and make override entries in the **values.yaml** file to enable the storage.

```
s3_storage_enabled: true
s3_storage:
  account:
    accessKey:  # The access key created when you created the storage bucket
    secretKey:  # The secret created when you created the bucket
    region: # The region in which the bucket was created
    entryPoint: # The URL for accessing bucket storage
gim_export:
  output_directory: # The bucket name
```

The `s3_storage` parameters are used to construct the **s3_storage_secrets** secret.

GKE example

```
gim_export:
  ...
  output_directory:                       "test-example-bucket-one"
  ...
s3_storage_enabled: true
s3_storage:
  account:
    accessKey: ""
    secretKey: ""
    region: ""
    entrypoint: "storage.googleapis.com"
```

Configure GIM behavior

You can specify values in the **values.yaml** file to control options that override aspects of the default configuration, thereby modifying GIM behavior and customizing the way data is stored in the Info Mart database. For information about the options you can configure including the default and valid values, see GIM configuration options.

To configure options, edit the GIM **values.yaml** file. Under the **gim_config** object in the, specify the

option and value in JSON format, noting the following:

- Options are separately configurable by tenant and, where applicable, by media type or even at the level of individual queues (DNs or scripts).
- Where an option can be configured at various levels, you can override a value set at a higher level (for example, for a particular media type in general) to set a different value for a particular lower-level object (for example, for that media type for an individual DN).
- See the note about configuration levels for information about the available configuration levels for certain options.

The entries in the **values.yaml** file are structured as follows:

```
gim_config: ""

log:
  level:                                "info"
  console_pattern_layout:               "%d{ISO8601} %-5p %-12t %m%n"
  appender:
    ConsoleLogger:
      Threshold:                         "info"

gim_etl:
  days_to_keep_active_facts:             "27"
  days_to_keep_deleted_annex:            "2"
  days_to_keep_discards_and_job_history: "60"
  days_to_keep_gidb_facts:               "27"
  days_to_keep_gim_facts:                "400"
  etl_start_date:                        ""
  max_chunks_per_job:                    "10"
  max_time_deviation:                    "30"
  memory_threshold:                      "0"
  partitioning_ahead_range:              "31"
  partitioning_interval_size_gidb:       "604800"
  partitioning_interval_size_gidb_mm:    "604800"
  partitioning_interval_size_gidb_ocs:   "604800"
  partitioning_interval_size_gim:        "2592000"
  purge_thread_pool_size:                "32"
  purge_transaction_size:                "100000"

date_time:
  date_time_max_days_ahead:              "400"
  date_time_min_days_ahead:              "183"
  date_time_start_year:                  "2020"
  date_time_tz:                          "GMT"
  first_day_of_week:                     "1"
  fiscal_year_start:                     ""
  fiscal_year_end:                       ""
  fiscal_year_week_pattern:              "none"
  min_days_in_first_week:                "1"
  simple_week_numbering:                 "true"

schedule:
  aggregate_duration:                    "23:00"
  aggregate_schedule:                    "30 0"
  etl_end_time:                          "23:30"
  etl_frequency:                         "1"
  etl_start_time:                        "00:00"
  export_schedule:                       "0/30 *"
  maintain_start_time:                   "23:40"
  run_aggregates:                        "true"
```

```
  run_export:                           "true"
  run_maintain:                         "true"
  run_scheduler:                        "true"
  run_update_stats:                     "true"
  on_demand_migration:                  "true"
  timezone:                             "UTC"
  update_stats_schedule:                "0/10 *"

gim_export:
  chunk_size_seconds:                   "86400"
  days_to_keep_output_files:            "30"
  max_retries:                          "3"
  output_directory:                     "gim-export"
  output_files_encoding:                "utf8"
  retry_delay_seconds:                  "30"
  start_date:                           ""
  thread_pool_size:                     "10"
  use_export_views:                     "true"
```

Custom calendars

GIM permits you to create custom calendars by creating a section in the values.yaml file that similar to the date-time section, and has the same options; name the section by using the *date-time-* prefix:

- **Job_InitializeGIM** populates data in all configured calendars when it initializes the Info Mart database.
- **Job_MaintainGIM** then maintains the calendars in accordance with options that are specified in the [date-time] and custom [date-time-*] configuration sections. The maintenance job automatically adjusts for special requirements such as daylight saving time (DST) and fiscal years that do not start on the same day every year (floating fiscal years).

Consider the settings for the **date-time** options carefully before the calendar dimension tables are populated for the first time. You can subsequently change the values of the **date-time-min-days-ahead** and **date-time-max-days-ahead** options at any time.

However, changing any of the other **date-time** options during runtime can introduce inconsistencies into the calendar data and affect reporting results adversely. For example, if you change the timezone option (**date-time-tz**) after Genesys Info Mart has been initialized, your reports can mix the results for different time zones within the same reporting interval.

Config Maps

Helm creates a number of Config Maps based on option values you specify in the **values.yaml** file (see Configure GIM Behavior). There are no Config Maps you can configure directly for Info Mart.

GIM configuration options

Contents

Options you can use to control GIM behavior.

Related documentation:

-
-
-
-

RSS:

- For private edition

GIM supports the configuration options described on this page. See Configure GIM behavior for information about how to change the option values in the Helm chart.

GIM behavior affects data in the Info Mart database dimensional model. For full information about the Info Mart database tables and columns referenced in the descriptions on this page, see the *Genesys Info Mart Historical Database Reference*.

- aggregate-duration
- aggregate-schedule
- appender.ConsoleLogger.Threshold
- chunk-size-seconds
- console-pattern-layout
- date-time-max-days-ahead
- date-time-min-days-ahead
- date-time-start-year
- date-time-tz
- days-to-keep-active-facts
- days-to-keep-deleted-annex
- days-to-keep-gidb-facts
- days-to-keep-gim-facts
- days-to-keep-output-files
- etl-end-time
- etl-frequency
- etl-start-date
- etl-start-time
- export-schedule
- first-day-of-week
- fiscal-year-end
- fiscal-year-start
- fiscal-year-week-pattern
- level
- maintain-start-time
- max-chunks-per-job
- max-retries
- max-time-deviation
- memory-threshold
- min-days-in-first-week
- on-demand-migration
- output-directory
- output-files-encoding
- partitioning-ahead-range
- partitioning-interval-size-gidb-mm
- partitioning-interval-size-gidb-ocs
- partitioning-interval-size-gidb
- partitioning-interval-size-gim
- purge-thread-pool-size
- purge-transaction-size
- retry-delay-seconds
- run-aggregates
- run-export
- run-maintain
- run-scheduler
- run-update-stats
- says-to-keep-discards-and-job-history
- simple-week-numbering
- start-date
- thread-pool-size
- timezone
- update-stats-schedule
- use-export-views

Important

When you specify an override value in the **values.yaml** file, you indicate the configuration level for that option value. The configuration level specifies the scope for which an option value is in effect. For example, for certain ***-threshold** options, you can specify different thresholds for the various media types, and you can further refine behavior by specifying different thresholds for individual queues. Only some options support configuration at different levels.

All options are configured at the tenant level. Below the tenant level, the available configuration levels are:

- standard — Applies across media types.
- media — Applies to the specified media type, such as voice or email.
- dn: media — Applies to the specified Virtual Queue DN. An option value specified on the DN level overrides a value specified for that option at the media level.
- script: media *or* script: standard — Applies to the specified Script object representing an Interaction Queue or Interaction Workbin for digital interactions. An option value specified on the Script level overrides a value specified for that option at the media or standard levels.

level

Default value: info
Valid values: debug, info, warn, error, none, all, off
Configuration level: standard

Determines whether local logging is enabled, and specifies the minimum level of events to log.

The DEBUG, INFO and WARN values correspond to the Genesys Management Layer DEBUG, TRACE, and STANDARD logging values, respectively.

>> Back to list

console-pattern-layout

Default value: %d{ISO8601} %-5p %-12t %m%n
Valid values: Any valid pattern string

Configuration level: standard

For information about this option, refer to the Apache logging site: http://logging.apache.org/index.html

>> Back to list

appender.ConsoleLogger.Threshold

Default value: info
Valid values: trace, debug, info, warn, error, all, off
Configuration level: standard

For information about this option, refer to the Apache logging site: http://logging.apache.org/index.html

>> Back to list

days-to-keep-active-facts

Default value: 27
Valid values: Any positive integer
Configuration level: standard

Specifies the maximum number of days to retain active multimedia interactions in GIDB and the dimensional model, including certain Staging tables, after which the interactions become eligible for purging. Depends on the value of **days-to-keep-gim-facts**.

>> Back to list

days-to-keep-deleted-annex

Default value: 2
Valid values: Any positive integer
Configuration level: standard

Specifies the number of days to retain deleted records in the *_ANNEX dimension tables.

The default value of days-to-keep-deleted-annex is small, because there is likely little reason to retain deleted data for significant periods of time. The major reason that Genesys Info Mart provides

*_ANNEX data is to support GCXI visibility controls, and GCXI uses only active *_ANNEX data for this purpose.

For example, with the default value (2 days), if Job_MaintainGIM is running on September 4, 2022, the job purges *_ANNEX records that terminated (in other words, the configuration setting on the object's Annex tab was deleted) before September 2, 2022.

>> Back to list

says-to-keep-discards-and-job-history

Default value: 60
Valid values: Any positive integer
Configuration level: standard

Specifies the number of days to retain data in the discard tables and audit and history tables.

The discard tables are Staging tables that store operational data that the transformation job was unable to process—for example, voice interaction data with unresolved IS-Links, or Configuration details records with missing configuration objects. The audit and history tables are Control tables that store information about data lineage and about ETL processing activity. Information in the discard, audit, and history tables is useful for troubleshooting.

Records in the discard, audit, and history tables are purged based on the timestamp of the ETL processing event—ETL_TS for the discard (Staging) tables and CREATED_TS for the audit and history (Control) tables.

For example, if Job_MaintainGIM is running on August 23, 2022 (day 235 of the year) and days-to-keep-discards-and-job-history=600, Job_MaintainGIM purges all records in the discard, audit, and history tables that were written by instances of the ETL jobs that ran before January 1, 2021 (day 1 of the previous year).

>> Back to list

days-to-keep-gidb-facts

Default value: 27
Valid values: Any positive integer
Configuration level: standard

Specifies the number of days to retain fact data in GIDB. Facts that have a start time that is earlier than the retention period are eligible to be purged. Related Options: days-to-keep-active-facts

The retention period for multimedia facts in GIDB is determined solely by the value of **days-to-keep-gidb-facts**.

Genesys expects the value of **days-to-keep-gidb-facts** to be less than 30. Although many GIDB tables contain personally identifiable information (PII), Genesys Info Mart does not include most GIDB tables in General Data Protection Regulation (GDPR) processing.

>> Back to list

days-to-keep-gim-facts

Default value: 400
Valid values: Any positive integer
Configuration level: standard

Specifies the number of days to retain fact data in the dimensional model. Facts that have a start time that is earlier than the retention period are eligible to be purged. Job_MaintainGIM does not purge active fact data, dimension data, or aggregate tables (if aggregation is enabled). Genesys Info Mart enforces that the value of **days-to-keep-active-facts**' ***be less than the value of*** *days-to-keep-gim-facts* ***and*** *days-to-keep-gidb-facts* ***be less than the value of*** *days-to-keep-gim-facts*.

Genesys Info Mart does not enforce a limit on the length of the retention period. However, Genesys strongly recommends against setting **days-to-keep-gim-facts**, and likewise **days-to-keep-active-facts**, to an excessively large value (for example, thousands of days). To avoid performance issues or job failures when Genesys Info Mart operates against an excessively large Info Mart database, be realistic about your requirements and available database resources, and apply best practices and recommendations from your RDBMS vendor. If you choose a very long retention period, you must carefully choose and tune the storage used for the Info Mart database; also carefully consider settings for partition size and other database- and performance-related configuration options, as well as parameters on the RDBMS side.

By definition, all voice interaction data in the dimensional model relates to completed interactions.

How Purge Options Work

To illustrate how various days-to-keep-* options combine to determine when data is purged, the following tables provide examples of different scenarios for GIDB and the dimensional model, respectively.

Purge examples

GIDB

Scenario	Start Time of Facts to Be Purged
• **days-to-keep-gidb-facts**=14. • **days-to-keep-active-facts**=30. • Job_MaintainGIM is running on December 4, 2020, after the last ETL cycle for the day. In other words:	November 21, 2020

Scenario	Start Time of Facts to Be Purged
• **days-to-keep-gidb-facts** threshold is November 21, 2020 • **days-to-keep-active-facts** threshold is November 5, 2020	

Dimensional Model

Scenario	Start Time of Facts to Be Purged

• **days-to-keep-gim-facts**=400.
• **days-to-keep-active-facts**=30.
• Job_MaintainGIM is running on December 4, 2020, after the last ETL cycle for the day. In other words:
  • **days-to-keep-gim-facts** threshold is November 1, 2019
  • **days-to-keep-active-facts** threshold is November 5, 2020
	November 1, 2019, or earlier
	The oldest timestamp for artificially terminated multimedia interactions is November 5, 2020.

>> Back to list

etl-start-date

Default value:
Valid values: Any date after 1970 in the format yyyy-mm-dd hh:mm:ss
Configuration level: standard

Specifies the earliest date for which Genesys Info Mart considers IDB data for extraction in a new deployment or when the Info Mart database is re-initialized. IDB data that has timestamps earlier than the ETL start date is never extracted.

The option is used only when Job_InitializeGIM initializes the database. If the **etl-start-date option** is not specified, the earliest starting point for Genesys Info Mart processing is IDB data that has timestamps 30 days prior to the Info Mart database initialization.

The main purpose of the option is to pre-empt performance and maintenance issues when Genesys Info Mart is introduced into a deployment with much older existing IDB data. Specifying the ETL start date enables users to:

- Shorten the backlog of IDB data if they do not need to include all of the old data in their reporting.
- In deployments with a partitioned Info Mart database, specify the starting point for creating partitions, which is important for proper partitioning.

Genesys Info Mart ignores this option value for Configuration details; after the database has been initialized or re-initialized, Genesys Info Mart extracts all cfg data going back as far as 2010, regardless of the value of **etl-start-date**.

>> Back to list

max-chunks-per-job

Default value: 10
Valid values: 1-100
Configuration level: standard

Specifies the number of extracted data chunks that the transformation job processes in one ETL cycle. As long as it seems practical from the performance perspective, increase the option value to transform a larger amount of data in a single cycle.

In deployments using the Data Export feature, the option also controls the maximum number of export chunks that can be created by the export job during each job run. In situations where the time range to be exported exceeds the export chunk size (for example, if there is a backlog because data is being re-exported), a larger number of export chunks reduces the time required to process a large export backlog.

>> Back to list

max-time-deviation

Default value: 30
Valid values: 1-120
Configuration level: standard

Specifies the maximum time deviation, in seconds, to consider for time synchronization inaccuracy between host-system clocks. This option also specifies the maximum acceptable delay the transformation job accommodates in scenarios in which ICON creates delayed records in the GM_F_USERDATA table in IDB.

If the maximum time deviation or delay is exceeded, Genesys Info Mart results become unreliable.

Genesys recommends the following relationship between the value of this option and the Advanced Disconnect Detection Protocol (ADDP) timeout, which is the Local Timeout parameter configured for ADDP connections to data sources on the Connections tab of the ICON Application: `[(ADDP Local Timeout) * 2] + (actual maximum difference in time synchronization between hosts)`

In HA deployments, Genesys Info Mart uses max-time-deviation for reliable analysis of the “no data” information in IDB; (`NoData − max-time-deviation`) is considered to be reliable for all data sources for a particular ICON provider.

>> Back to list

memory-threshold

Default value: 0
Valid values: 0-99
Configuration level: standard

Specifies the percentage of available memory that must be exceeded before Genesys Info Mart logs a message (55-20101) that indicates that the memory threshold has been exceeded. If the value of this option is 0, the feature is disabled.

>> Back to list

partitioning-ahead-range

Default value: 31
Valid values: Any positive integer
Configuration level: standard

Specifies, in terms of number of days, how far ahead Job_InitializeGIM (in the first instance) and Job_MaintainGIM (on an ongoing basis) create partitions for GIDB, Control, and Info Mart fact tables that are partitioned. These jobs add partitions for the number of days ahead of the time that the job is running. (Job_InitializeGIM also adds partitions from etl-start-date up to the time that the job is running.) The number of partitions that Job_MaintainGIM actually creates during each run depends on the partition sizes and the job frequency.

Example

If `partitioning-interval-size-gidb=86400` (one day), `partitioning-interval-size-gim=604800` (one week), and `partitioning-ahead-range=14`, Job_MaintainGIM creates as many other partitions as necessary to provide partitions for GIDB, Control, and Info Mart fact tables up to 14 days ahead. If Job_MaintainGIM runs daily, then:

- For GIDB tables, each run of Job_MaintainGIM creates one new partition of size 1 day, for the fourteenth day. (Previous runs have created partitions for the other days.)
- For Info Mart fact tables and Control tables, the maintenance job creates one new partition of size 7 days at the start of each week. (A previous run will have created a partition for the other week.)

  If the value of **partitioning-ahead-range** is not a multiple of **partitioning-interval-size-gim,** the maintenance job creates a new partition only when the last day of the partitioning ahead range falls in a week for which a partition has not been created. For example, if `partitioning-interval-size-gim=604800 (7 days)` but `partitioning-ahead-range=10`, two new partitions are created on the first run of Job_MaintainGIM, and the next partition is created on the fifth run.

To guarantee that partitions are always available for use by the ETL, ensure that run-scheduler is set to `true` (the default value).

This option applies only in deployments that use partitioning.

>> Back to list

partitioning-interval-size-gidb

Default value: 604800
Valid values: Any positive integer
Configuration level: standard

Specifies the size of partitions, in seconds, for GIDB tables that are partitioned. Job_MaintainGIM creates partitions of the specified size in the Info Mart database in preparation for future ETL cycles. The default size of GIDB table partitions is 24 hours (86400 seconds).

In PostgreSQL deployments, Genesys recommends setting the size of GIDB table partitions to one week (604800 seconds).

This option applies only in deployments that use partitioning.

>> Back to list

partitioning-interval-size-gidb-mm

Default value: 604800
Valid values: Any positive integer
Configuration level: standard

Specifies the size of partitions, in seconds, for partitioned GIDB tables that store multimedia interaction data. When this option is set, Job_MaintainGIM creates partitions of the specified size in the Info Mart database in preparation for future ETL cycles. If the option is not specified, the value of partitioning-interval-size-gidb, which has a default value of 24 hours (86400 seconds), is used.

Genesys recommends increasing the size of GIDB partitions for multimedia interactions, which typically live longer than voice interactions but generate a smaller volume of data.

In PostgreSQL deployments, Genesys recommends setting the size of GIDB table partitions to one week (604800 seconds).

Transformation performance is optimal when partitions are large enough that data that is being actively used is in the fewest number of partitions, and the partitions are small enough that their indexes can fit into the cache.

>> Back to list

partitioning-interval-size-gidb-ocs

Default value: 604800
Valid values: Any positive integer
Configuration level: standard

Specifies the size of partitions, in seconds, for partitioned GIDB tables that store Outbound Contact–related data. When this option is set, Job_MaintainGIM creates partitions of the specified size in the Info Mart database in preparation for future ETL cycles. If the option is not specified, the value of partitioning-interval-size-gidb, which has a default value of 24 hours (86400 seconds), is used.

In PostgreSQL deployments, Genesys recommends setting the size of GIDB table partitions to one week (604800 seconds).

Transformation performance is optimal when partitions are large enough that data that is being actively used is in the fewest number of partitions, but small enough that their indexes can fit into the cache.

>> Back to list

partitioning-interval-size-gim

Default value: 2592000
Valid values: Any positive integer
Configuration level: standard

Specifies the size of partitions, in seconds, for Info Mart fact and Control tables that are partitioned. Job_MaintainGIM creates partitions of the specified size in the Info Mart database in preparation for future ETL cycles.

In PostgreSQL deployments, the recommended size of partitions for dimensional-model data depends on your plans for data retention in the Info Mart database. For PostgreSQL, Genesys recommends setting the size of fact table partitions to:

- One month (2592000 seconds) if data retention is under three years (**days-to-keep-gim-facts** is less than 1095)
- Two or three months (5184000 or 7776000 seconds) if data retention is more than three years (**days-to-keep-gim-facts** is greater than 1095).

This option applies only in deployments that use partitioning.

>> Back to list

purge-thread-pool-size

Default value: 32
Valid values: Any positive integer
Configuration level: standard

Specifies the maximum number of concurrent purging transactions. The optimum value for this option depends on the characteristics and capacity of your deployment. Consider increasing the value of this option if you think that there is scope to improve performance of the purge operation.

>> Back to list

purge-transaction-size

Default value: 100000
Valid values: Any positive integer
Configuration level: standard

Specifies the number of deleted records per table that are committed in a single transaction.

For example:

- If there are 150,000 records in a particular table that are eligible for purging and **purge-transaction-size**=`100000`, Job_MaintainGIM deletes and commits 100,000 records in one transaction and 50,000 records in a separate transaction.
- If there are 90,000 records in one table, 10,000 records in another table, and **purge-transaction-size**=`100000`, Job_MaintainGIM deletes and commits 90,000 records from the first table in one transaction and 10,000 records from the second table in a separate transaction.

>> Back to list

date-time-max-days-ahead

Default value: 400
Valid values: Any positive integer
Configuration level: standard

Specifies, in number of days, how far ahead the calendar dimension table is populated. The default value specifies that the calendar dimension is populated up to a year in advance (365 days + 1 day

for leap years). Genesys does not recommend that you populate the calendar tables more than a year in advance, in case there are changes to DST or other international time standards that might invalidate the prepopulated data.

Note: Ensure that you populate the calendar far enough ahead to meet the requirements of your reporting intervals.

>> Back to list

date-time-min-days-ahead

Default value: 183
Valid values: Any positive integer
Configuration level: standard

Specifies, in number of days that remain in the prepopulated calendar, when the calendar table is updated with the next batch of days ahead. The default value specifies that the maintenance job updates this calendar approximately 6 months before it expires.

>> Back to list

date-time-start-year

Default value: 2020
Valid values: 1970-2038
Configuration level: standard

Specifies the year that the calendar starts. When you are setting this option, ensure that you choose a start year that provides sufficient buffer to prevent inconsistencies or unexpected missing dimensions around the start of the calendar. Genesys recommends that you set the value so that the calendar starts at least one year prior to any date that might be encountered in the data. Be aware that Genesys Info Mart uses GMT for internal time references, and this affects exactly when the calendar starts.

For example, if the other **[date-time]** options that affect the start date are set so that the calendar starts at 00:00 AM on January 1, 2022, and the **date-time-tz** option is set to Eastern European Time (GMT + 2), the calendar table is populated with dimensions starting at 02:00 AM on January 1, 2022.

>> Back to list

date-time-tz

Default value: GMT
Valid values: Any valid Java time zone
Configuration level: standard

Specifies the time zone for the calendar. You can use any valid time zone that is supported by the version of the Java Runtime Environment (JRE) that runs the Genesys Info Mart Server. For more information about supported time zones, see the documentation about calendar time zones on the Java developer website or other public resources.

Sample public resources:

- http://www.java2s.com/Tutorial/Java/0120__Development/GettingallthetimezonesIDs.htm
- http://en.wikipedia.org/wiki/Zone.tab

> **Important**
> Particularly in deployments that use GCXI or RAA, ensure that the time zone is set appropriately for your deployment before you initialize Genesys Info Mart or before aggregation starts.

>> Back to list

first-day-of-week

Default value: 1
Valid values: 1-7 (Sunday-Saturday)
Configuration level: standard

Specifies the day of the week that is considered to be the start of the week. For example, 1 (Sunday) is usually the first day of the week in the United States; for countries that use the ISO 8601 standard, 2 (Monday) is the first day of the week.

>> Back to list

fiscal-year-start

Default value:
Valid values: Any valid combination of month and day, in M-d format

Configuration level: standard

Specifies the month and day that the fiscal year starts. For example, 1-1 means January 1; 10-1 means October 1. Set either this value, or **fiscal-year-end**, but not both.

If simple-week-numbering=true, every fiscal year starts on the fixed date that is specified by this option. If **simple-week-numbering**=`false`, the fiscal year starts on the first day of the week that contains the date that is specified by this option; however, the actual start date depends on the value of the **first-day-of-week** option. Genesys Info Mart adjusts automatically for the floating fiscal year. For example, if **simple-week-numbering**=`false`, **fiscal-year-start**=`3-1`, and **first-day-of-week**=1, then:

- Fiscal year 2012 starts on February 26.
- Fiscal year 2013 starts on February 24.
- Fiscal year 2014 starts on February 23.

This option requires that **fiscal-year-week-pattern** be set to a valid pattern.

>> Back to list

fiscal-year-end

Default value:
Valid values: Any valid combination of month and day, in M-d format.
Configuration level: standard

Specifies the month and day that the fiscal year ends. For example, 1-31 means January 31; 11-30 means November 30. Set either this value, or **fiscal-year-start**, but not both.

For example, to have calendar date 2022-11-01 in fiscal year 2023, set no value for **fiscal-year-start**, and set **fiscal-year-end**=`10-31`. This option requires that **fiscal-year-week-pattern** be set to a valid pattern.

>> Back to list

fiscal-year-week-pattern

Default value: none
Valid values: none, 544, 454, 445
Configuration level: standard

Specifies the pattern for the number of weeks in each month of a fiscal quarter. For example, 544 means 5 weeks in the first month, 4 weeks in the second month, and 4 weeks in the third month of

each quarter. A value of none means that the calendar is not a fiscal one.

>> Back to list

min-days-in-first-week

Default value: 1
Valid values: 1-6
Configuration level: standard

Specifies the minimum number of days from the new year that must be in the first week of the year, if simple week numbering is not used and there are no partial weeks in the calendar year. The ISO 8601 standard does not use simple week numbering.

The ISO 8601 definition of the first week in the year is the week that has the first Thursday in it. To conform to the ISO 8601 standard, set simple-week-numbering=false, first-day-of-week=2, and min-days-in-first-week=4.

For example, if simple-week-numbering=false, first-day-of-week=2, and January 1 of the new year is on a Friday, there are 3 days from the new year in the week that starts on Monday, December 28. Therefore:

- If the value of this option is set to 1, the calendar counts the first week of the new year as starting on Monday, December 28.
- If the value of this option is set to 4, the week that starts on Monday, December 28, is assigned to the previous year, and the calendar counts the first week of the new year as starting on Monday, January 4.

>> Back to list

simple-week-numbering

Default value: true
Valid values:
Configuration level: standard

Specifies whether the calendar year and the week-numbering year coincide.

For simple week numbering, Week 1 always begins on the first day of the calendar year (for Gregorian calendars, January 1; for fiscal calendars, the day that is specified in the fiscal-year-start option). As a result, the first and last weeks of the year might be partial weeks, because the first week does not necessarily start with the day that is specified by the first-day-of-week option. To comply with ISO 8601 week numbering, set the value of this option to false.

>> Back to list

aggregate-duration

Default value: 23:00
Valid values: 00:00-24:00
Configuration level: standard

Specifies the amount of time, in 24-hour format, that Job_AggregateGIM runs after it is launched. When the run-aggregates option is set to TRUE, the scheduler stops the aggregation job when this interval expires. The aggregation job is launched in accordance with a schedule defined by the aggregate-schedule option. After the aggregation job is launched, it runs continuously until the aggregation-duration interval expires.

>> Back to list

aggregate-schedule

Default value: 30 0
Valid values: A valid CRON expression
Configuration level: standard

Specifies the daily schedule for Job_AggregateGIM to start. The job starts in accordance with this schedule when aggregation is being controlled by the scheduler (in other words, the run-aggregates option is set to true). Between them, the **aggregate-schedule** and **aggregate-duration** options define daily time intervals within which Job_AggregateGIM runs continuously.

The schedule is defined in the format of a CRON expression that represents a set. The expression comprises two fields, which are separated by whitespace:

- The first field specifies minutes. Valid values are 0–59 and optional special characters (see below).
- The second field specifies hours. Valid values are 0–23 and allowed special characters.

The following special characters are allowed in the CRON expression:

- , (comma)—Separates items in a list. For example, specifying the first field (minutes) as 0,30,45 means the 0th, 30th, and 45th minutes of the hour.
- - (hyphen)—Defines a range. For example, specifying the first field (minutes) as 30-35 means every minute between the 30th and 35th minute of the hour, inclusive; this is the same as specifying 30,31,32,33,34,35.
- * (asterisk)—Indicates that the CRON expression matches for all values of the field. For example, specifying the second field (hours) as * means every hour in the day.
- / (forward slash)—Describes increments. For example, specifying the first field (minutes) as 0/10 means the 0th minute of the hour and every 10 minutes thereafter.

Examples

The following values for **aggregate-schedule** illustrate sample schedules:

- `0 1` means that the aggregation job launches once a day at 01:00.
- `30 0,3/2` means that the aggregation job launches every day at 00:30, 03:30, and every 2 hours after that for the rest of the day.
  This schedule assumes that the value of **aggregate-duration** is `02:00` or less. The scheduler does not launch a new instance of Job_AggregateGIM while an existing instance is running. For aggregation to run on the specified schedule, the value of **aggregate-duration** must not exceed the intervals between scheduled start times.
- `30 *` means that the aggregation job launches every hour during the day on the half-hour (00:30, 01:30, 02:30, and so on), assuming that the value of **aggregate-duration** is `01:00` or less.

Genesys recommends against configuring a schedule that has the aggregation job running in a series of short bursts—for example, **aggregate-schedule**=`30 *` and **aggregate-duration**=`00:15`. When the time specified by **aggregate-duration** expires, the scheduler immediately stops the aggregation job, even if it is in the middle of processing a batch of data.

If you want Job_AggregateGIM to run continuously for 24 hours a day, without any breaks for maintenance activities (which is not recommended), set **aggregate-schedule**=`0 0` and **aggregate-duration**=`24:00`.

>> Back to list

etl-end-time

Default value: 23:30
Valid values: 00:00-23:59
Configuration level: standard

Specifies the time of day, in 24-hour format, when the last ETL cycle can start running. If the value that you specify is before the ETL start time, the end time is for the next day (past midnight).

If etl-start-time=etl-end-time, the ETL cycle runs continuously.

Ensure that you configure **etl-start-time** and **etl-end-time** so that there is sufficient time for the last ETL cycle of the day to complete and for Job_MaintainGIM to run (see the **maintain-start-time** option), before the start of the first ETL cycle of the next day.

>> Back to list

etl-frequency

Default value: 1
Valid values: 0-1440
Configuration level: standard

Specifies the number of minutes that pass between the start times of each ETL cycle. If the amount of time that it takes to complete a cycle is shorter than the specified value, the next cycle is delayed until the time elapses. If the amount of time that it takes to complete a cycle is longer than the specified value, the next cycle is started immediately.

The ETL frequency must not be greater than the chunk size for data extraction, as specified by the **extract-data-chunk-size** option. Otherwise, Genesys Info Mart can not keep pace with ICON. When it checks the deployment, Genesys Info Mart verifies the internal consistency between the ETL frequency and extraction chunk size.

By default, the value of **etl-frequency** is much smaller than the value of **extract-data-chunk-size**. Genesys recommends that you retain this relationship, to minimize data latency. For example, say that **extract-data-chunk-size**=900 (15 minutes), **etl-frequency**=1, and all data from the last chunk has been processed; when the next ETL cycle starts 1 minute later, there is only 1 minute's worth of new data, and this can be processed very quickly. Alternatively, if there is a backlog of data, and it takes less than 15 minutes to process a 15-minute chunk, the next ETL cycle starts almost immediately, to continue catching up.

>> Back to list

etl-start-time

Default value: 00:00
Valid values: 00:00-23:59
Configuration level: standard

Specifies the time of day, in 24-hour format, when the first ETL cycle starts running.

>> Back to list

export-schedule

Default value: 0/30
Valid values: A valid CRON expression
Configuration level: standard

Defines the time intervals at which Job_ExportGIM runs. The job starts and runs periodically in accordance with this schedule when the **run-export** option is set to **true**. By default, the job runs at

00:20, 08:20, and 16:20 every day.

The default schedule, run in conjunction with the default **chunk-size-seconds** option in the **[gim-export]** section, is designed to keep daily disruptions or delays from carrying over to the next day.

Job_ExportGIM can run in conjunction with the ETL jobs, but not in conjunction with Job_MaintainGIM.

The schedule is defined in the format of a CRON expression that represents a set. The expression comprises two fields, which are separated by whitespace:

- The first field specifies minutes. Valid values are 0–59 and optional special characters (see below).
- The second field specifies hours. Valid values are 0–23 and allowed special characters.

The following special characters are allowed in the CRON expression:

- , (comma)—Separates items in a list.
- - (hyphen)—Defines a range.
- * (asterisk)—Indicates that the CRON expression will match for all values of the field.
- / (forward slash)—Describes increments.

>> Back to list

maintain-start-time

Default value: 23:40
Valid values: 00:00-23:59
Configuration level: standard

Specifies the time of day, in 24-hour format, when Job_MaintainGIM is started. This job is scheduled to start at this time when the run-maintain option is set to TRUE. The value that you specify must be outside the range that is specified by etl-start-time and etl-end-time.

> **Tip**
> If the time of day that is represented by the new value has already passed, the new value is applied to the following day.

>> Back to list

run-aggregates

Default value: true
Valid values: true, false
Configuration level: standard

Specifies whether the scheduler manages the aggregation job, to run the aggregation engine inside the Genesys Info Mart process.

- When the value of this option is set to `true`, the scheduler starts Job_AggregateGIM at the scheduled time, as specified by the aggregate-schedule option; Job_AggregateGIM then runs continuously until the scheduler stops the job after the scheduled interval, as specified by the aggregate-duration option. The scheduler does not allow a second aggregation process to be launched while the job is running. The scheduler also does not allow any other aggregation process to be launched outside the intervals that are defined by the **aggregate-schedule** and **aggregate-duration** options.
- When the value of this option is set to `false`, the scheduler does not manage the aggregation job at all, leaving aggregation in whatever state it is in when the option value is set. This means that, if you change the value of **run-aggregates** to `false` while the aggregation job is running, the scheduler never stops the job.

For example, if **run-aggregates**=`true`, **aggregate-schedule**=`0 1`, and **aggregate-duration**=`05:00`, the aggregation job runs continuously between 01:00 AM and 06:00 AM daily. The scheduler does not allow you to launch a second instance of Job_AggregateGIM manually from the management GUI (Genesys Info Mart Manager or the Genesys Info Mart Administration Console) within that time period. Furthermore, if you try to launch an instance of Job_AggregateGIM manually from the management GUI outside that time period (for example, at 08:00 AM), the scheduler identifies that the job is not supposed to be running at that time and stops it. If you want to run Job_AggregateGIM manually from the management GUI outside the scheduled times, you must first set **run-aggregates** to `false`.

>> Back to list

run-export

Default value: true
Valid values: true, false
Configuration level: standard

Specifies whether Job_ExportGIM runs. When the value of this option is set to true, the scheduler starts and runs the job at the time and intervals specified by the **export-schedule** option.

>> Back to list

run-maintain

Default value: true
Valid values: true, false
Configuration level: standard

Specifies whether to run Job_MaintainGIM at the scheduled time, as specified by the maintain-start-time option.

>> Back to list

run-scheduler

Default value: true
Valid values: true, false
Configuration level: standard

Specifies whether to stop or start the scheduler. If the value of this option was set to TRUEso that the scheduler is scheduling jobs, and you change the value of this option to FALSE, the scheduler pauses, with no effect on any jobs that might already be running. If you then reset the value to TRUE, the scheduler resumes at the point at which it stopped.

>> Back to list

run-update-stats

Default value: true
Valid values: true, false
Configuration level: standard

Specifies whether Job_UpdateStats runs in PostgreSQL deployments, at the time and intervals specified by the **update-stats-schedule** option.

>> Back to list

on-demand-migration

Default value: true
Valid values: true, false

Configuration level: standard

Controls whether Genesys Info Mart runs Job_MigrateGIM automatically if the Info Mart database schema is not up to date following migration of the Info Mart server.

- true — Genesys Info Mart launches Job_MigrateGIM automatically if the schema is not up to date.
- false — Genesys Info Mart dpes not launch Job_MigrateGIM automatically if the schema is not up to date and Genesys Info Mart enters the migration state.

> **Important**
>
> Genesys does not recommend enabling migration on demand unless policies and procedures are in place to ensure that essential pre-migration and post-migration steps are also performed without manual intervention — for example, frequent database backup and re-creation of read-only views following migration.

The value of `false` preserves previous Genesys Info Mart behavior, which requires you to run Job_MigrateGIM manually before ETL functioning resumes if Genesys Info Mart has entered the migration state.

Enabling on-demand migration is suitable only if you always want to apply schema updates immediately, without review and without controlling the timing of the schema update or required system preparation.

>> Back to list

timezone

Default value: UTC
Valid values:
Configuration level: standard

Specifies the time zone in which the schedule is defined. Internally, Genesys Info Mart maintains the schedule in UTC time. For convenience, you can use this option to specify a local time zone that makes it easier for you to plan and manage the schedule. You can use any valid time zone that is supported by the version of the JRE that runs the Genesys Info Mart Server.

For more information about supported time zones, see the documentation about calendar time zones on the Java developer website or other public resources. For sample reference sites, see the description of the **date-time-tz** option.

>> Back to list

update-stats-schedule

Default value: 0/10
Valid values: A valid CRON expression
Configuration level: standard

Defines the time intervals at which Job_UpdateStats runs. The job starts and then runs periodically in accordance with this schedule. By default, the job runs every 10 minutes throughout the day. Job_UpdateStats can run in conjunction with the ETL jobs, but not in conjunction with Job_MaintainGIM.

The schedule is defined in the format of a CRON expression that represents a set. The expression comprises two fields, which are separated by white space:

- The first field specifies minutes. Valid values are 0–59 and optional special characters (see below).
- The second field specifies hours. Valid values are 0–23 and allowed special characters.

The following special characters are allowed in the CRON expression:

- , (comma)—Separates items in a list. For example, specifying the first field (minutes) as 0,30,45 means the 0th, 30th, and 45th minutes of the hour.
- - (hyphen)—Defines a range. For example, specifying the first field (minutes) as 30-35 means every minute between the 30th and 35th minute of the hour, inclusive; this is the same as specifying 30,31,32,33,34,35.
- * (asterisk)—Indicates that the CRON expression will match for all values of the field. For example, specifying the second field (hours) as * means every hour in the day.
- / (forward slash)—Describes increments. For example, specifying the first field (minutes) as 0/10 means the 0th minute of the hour and every 10 minutes thereafter.

The schedule that you configure for Job_UpdateStats does not need to specifically allow for a maintenance window: A running instance of Job_UpdateStats does not prevent Job_MaintainGIM from starting, and once Job_MaintainGIM has started as part of the schedule, the scheduler suspends the schedule for Job_UpdateStats until Job_MaintainGIM finishes.

For values that illustrate sample schedules, see the examples for the aggregate-schedule option.

>> Back to list

chunk-size-seconds

Default value: 86400
Valid values: Any positive integer
Configuration level: standard

Specifies the size of the time interval, in seconds, for which data is exported in each job.

>> Back to list

days-to-keep-output-files

Default value: 30
Valid values: Any positive integer
Configuration level: standard

Specifies how many days to store exported files before deleting them.

>> Back to list

max-retries

Default value: 3
Valid values: Any non-negative integer
Configuration level: standard

Specifies the maximum numbers of retries the job does in the case of intermittent failures before failing the job.

>> Back to list

output-directory

Default value: gim-export
Valid values: Valid directory path (might not exist)
Configuration level: standard

Specifies the directory where exported files are stored.

>> Back to list

output-files-encoding

Default value: utf8
Valid values: Character encoding supported by Java
Configuration level: standard

Specifies the character encoding for exported files. For Java-supported encodings, see Supported Encodings.

>> Back to list

retry-delay-seconds

Default value: 30
Valid values: Any non-negative integer
Configuration level: standard

Specifies the amount of time, in seconds, that the job waits (in the case of intermittent failure) before attempting to run again.

>> Back to list

start-date

Default value:
Valid values: Any date after 1970 in the format yyyy-mm-dd hh:mm:ss
Configuration level: standard

Specifies the earliest date for which Job_ExportGIM exports data, and then continues with incremental exports. If no date is specified (the default), Job_ExportGIM starts exporting from the beginning of the Info Mart data. Requires that the value of **days-to-keep-discards-and-job-history** is greater than or equal to the value of **days-to-keep-gim-facts**.

- This option applies only when the directory in which Genesys Info Mart stores the exported files (the directory specified by the output-directory option) is empty.
- This option is not supported when audit log retention time is shorter than facts retention time.

>> Back to list

thread-pool-size

Default value: 10
Valid values: Any positive integer
Configuration level: standard

Specifies the maximum number of worker threads that are used to export data concurrently.

>> Back to list

use-export-views

Default value: true
Valid values: true, false
Configuration level: standard

Controls whether Genesys Info Mart exports data using export views, which represent a snapshot of the Info Mart schema at the time the export views were created.

- **false**—Genesys Info Mart exports data using the current schema. The export might include schema changes, such as new or renamed tables or columns, that occurred as part of a recent Info Mart migration.
- **true**—Genesys Info Mart exports data using export views. The export includes the same tables and columns as the previous export(s), regardless of any schema changes resulting from migrations that have occurred in the interim.

>> Back to list

Deploy GIM

Contents

Learn how to deploy GIM (GIM) into a private edition environment.

Related documentation:

-
-
-
-

RSS:

- For private edition

Assumptions

- The instructions on this page assume you are deploying the service in a service-specific namespace, named in accordance with the requirements on Creating namespaces. If you are using a single namespace for all private edition services, replace the namespace element in the commands on this page with the name of your single namespace or project.
- Similarly, the configuration and environment setup instructions assume you need to create namespace-specific (in other words, service-specific) secrets. If you are using a single namespace for all private edition services, you might not need to create separate secrets for each service, depending on your credentials management requirements. However, if you do create service-specific secrets in a single namespace, be sure to avoid naming conflicts.

> **Important**
>
> Make sure to review Before you begin GIM deployment for the full list of prerequisites required to deploy GIM, including creation of the Info Mart database and, if applicable, S3-compatible storage for your exported data (see (see Create object storage).

Set up your environment

To prepare your environment for the deployment, complete the steps in this section for:

- GKE
- AKS

GKE environment setup

1. Ensure that the gcloud CLI and required Helm version are installed on the host where you will run the deployment.
2. Using the CLI, log in to the GKE cluster from the host where you will run the deployment:

   ```
   gcloud container clusters get-credentials
   ```

3. If the cluster administrator has not already done so, create a new namespace `gim` for GIM:
   - Create a JSON file specifying the namespace metadata. For example, **create-gim-namespace.json**:

     ```
     {
        "apiVersion": "v1",
        "kind": "Namespace",
        "metadata": {
          "name": "gim",
          "labels": {
            "name": "gim"
          }
        }
     }
     ```

   - Execute the following command to create the `gim` namespace:

     ```
     kubectl apply -f apply create-gim-namespace.json
     ```

   - Confirm namespace creation:

     ```
     kubectl describe namespace gim
     ```

4. Create the pull secret for the image registry.
   - This step defines a secret so that Kubernetes can authenticate your image repository and pull artifacts from it. The repository is represented as `docker-registry` in the system. For information about downloading artifacts from the repository, see Downloading your Genesys Multicloud CX containers.
   - When you configure the GIM service, you will reference the registry pull secret as a Helm chart override; see GIM Helm chart overrides.

     ```
     kubectl create secret docker-registry  --docker-server= --docker-username= --docker-
      password= --docker-email= -n gim
     ```

AKS environment setup

1. Ensure that the Azure CLI and required Helm version are installed on the host where you will run the deployment.
2. Using the CLI, log in to the Azure cluster from the host where you will run the deployment.

   ```
   az aks get-credentials --resource-group  --name  --admin
   ```

3. If the cluster administrator has not already done so, create a new namespace `gim` for GIM:
   - Create a JSON file specifying the namespace metadata; for example, **create-gim-namespace.json** as in the following example:

```
{
   "apiVersion": "v1",
   "kind": "Namespace",
   "metadata": {
     "name": "gim",
     "labels": {
       "name": "gim"
     }
   }
}
```

- Execute the following command to create the `gim` namespace:

  ```
  kubectl apply -f apply create-gim-namespace.json
  ```

- Confirm namespace creation:

  ```
  kubectl describe namespace gim
  ```

4. Create the pull secret for the image registry.
   - This step defines a secret so that Kubernetes can authenticate your image repository and pull artifacts from it. The repository is represented as `docker-registry` in the system. For information about downloading artifacts from the repository, see Downloading your Genesys Multicloud CX containers.
   - When you configure the GIM service, you will reference the registry pull secret as a Helm chart override; see GIM Helm chart overrides.

     ```
     kubectl create secret docker-registry  --docker-server= --docker-username= --docker-
      password= --docker-email= -n gim
     ```

Deploy

Execute the following command to install GIM:

```
helm upgrade --install gim  -f  -n gim
```

Execute the following command to install GIM monitoring:

```
helm upgrade --install gim-monitoring  -n gim
```

When the GIM service starts, it connects to the Info Mart database and checks if the GIM schema has been initialized. If the schema has not been initialized, the service runs a script to initialize the GIM schema.

Validate the deployment

You can consider GIM deployment successful when the pod is running and in Ready state. Genesys Info Mart does not report the Ready state for pods until internal health checks are satisfied and the pods are operational. For example, GIM does not report Ready until the first transformation job has completed successfully. In other words, if the GIM pod is running and in Ready state, it means that GIM successfully started, connected to Kafka and the Info Mart database, initialized the Info Mart

database, and completed the first ETL cycle.

You can use standard `kubectl` commands like `list` and `get` to verify the successful deployment and readiness status of the Kubernetes objects, including connection to the database.

However, from a functional point of view, you cannot validate deployment of the GIM service and database unless GCA and GSP have been deployed as well. Do not expect consistent data until all three Genesys Info Mart services are up and running. For more details about functional checks you can perform to validate GIM deployment, see the equivalent validation section on the Deploy GIM Stream Processor page.

Before you begin GCA deployment

Contents

Find out what to do before deploying GIM Config Adapter (GCA).

Related documentation:

-
-
-
-

RSS:

- For private edition

Limitations and assumptions

Instructions are provided for a single-tenant deployment.

Download the Helm charts

GIM Config Adapter (GCA) and GCA monitoring are the only services that run in the GCA Docker container. The Helm charts included with the GCA release provision GCA and any Kubernetes infrastructure necessary for GCA to run.

See Helm charts and containers for Genesys Info Mart for the Helm chart versions you must download for your release.

For information about how to download the Helm charts, see Downloading your Genesys Multicloud CX containers.

Third-party prerequisites

For information about setting up your Genesys Multicloud CX private edition platform, see Software requirements.

The following table lists the third-party prerequisites for GCA.

Third-party services

Name	Version	Purpose	Notes
Kafka	2.x	Message bus.	GCA publishes configuration data to the `gca-cfg` topic, which GSP consumes. The topic must exist in your Kafka configuration.
Object storage		Persistent or shared data storage, such as Amazon S3, Azure Blob Storage, or Google Cloud Storage.	Both GCA and GSP require object storage to store data during processing. You can use the same storage account for both services.
A container image registry and Helm chart repository		Used for downloading Genesys containers and Helm charts into the customer's repository to support a CI/CD pipeline. You can use any Docker OCI compliant registry.	
Command Line Interface		The command line interface tools to log in and work with the Kubernetes clusters.	

Storage requirements

GCA uses object storage to store the GCA snapshot during processing. Like GSP, GCA supports using S3-compatible storage provided by OpenShift and Google Cloud Platform (GCP), and Genesys expects you to use the same storage account for GSP and GCA. If you want to use separate storage for GCA, follow the Configure S3-compatible storage instructions for GSP to create similar S3-compatible storage for GCA.

Network requirements

No special network requirements.

Browser requirements

Not applicable

Genesys dependencies

- Voice Tenant Service, which enables GCA to access the Configuration Server database. You must deploy the Voice Tenant Service before you deploy GCA.
  - Ensure that an appropriate user account is available for GCA to use to access the Configuration Database. The GCA user account requires at least read permissions.
  - You must also have your Tenant ID information available.
- There are no strict dependencies between the Genesys Info Mart services, but the logic of your particular pipeline might require Genesys Info Mart services to be deployed in a particular order. Depending on the order of deployment, there might be temporary data inconsistencies until all the Genesys Info Mart services are operational. For example, GSP looks for the GCA snapshot when it starts; if GCA has not yet been deployed, GSP will encounter unknown configuration objects and resources until the snapshot becomes available.

For detailed information about the correct order of services deployment, see Order of services deployment.

GDPR support

Not applicable. GCA does not store information beyond an ephemeral snapshot.

Configure GCA

Contents

Learn how to configure GIM Config Adapter (GCA).

Related documentation:

-
-
-
-

RSS:

- For private edition

GCA Helm chart overrides

The GCA requires some configuration for deployment that must be done by modifying the GCA's default Helm chart. You do this by creating override entries in the GCA's **values.yaml** file.

Download the **gca** and **gca-monitoring** Helm charts from your image registry, using the appropriate credentials.

For information about how to download the Helm charts, see Downloading your Genesys Multicloud CX containers. To find the correct Helm chart version for your release, see Helm charts and containers for Genesys Info Mart. For general information about Helm chart overrides, see Overriding Helm chart values in the *Genesys Multicloud CX Private Edition Guide*.

At minimum, you must create entries in the **values.yaml** file to specify key system information, as described in the following sections.

> **Important**
>
> Treat your modified **values.yaml** file as source code, which you are responsible to maintain so that your overrides are preserved and available for reuse when you upgrade.

Image registry and pull secret

Image registry

Create an entry in the GSP's **values.yaml** file to specify the location of the Genesys JFrog image registry. This is the repository from which Kubernetes will pull images.

The location of the Genesys JFrog image registry is defined when you set up the environment for the GSP. It is represented in the system as the `docker-registry`. In the GSP Helm chart, the repository is represented as `image: registry`, as shown below. You can optionally set a container version for the image.

```
image: # The repository from which Kubernetes will pull images
  registry:  # The default registry is pureengage-docker-staging.jfrog.io
  tag:  # The container image tag/version
```

Pull secret

When you set up your environment, you provision a pull secret for Genesys JFrog image registry (`docker-registry`). Each service must supply the credentials for the repository in order for Kubernetes to be able to pull from the repository. Each of the three Info Mart services (GIM, GSP, and GCA) must be configured with the pull secret. You do this **values.yaml** for the service.

```
imagePullSecrets:
  docker-registry: {} # The credentials Kubernetes will use to pull the image from the
registry
```

Note that other services use a different syntax than this to configure the repository pull secret, as follows:

```
imagePullSecrets:
  name: docker-registry
```

Genesys Info Mart, GIM Stream Processor, and GIM Configuration Adaptor helm charts all support advanced templating that allow the helm to create the pull secret automatically; hence the variation in syntax.

Tenant ID

The Tenant microservice provides direct access to the configuration server database. Configure a Helm override entry for the Tenant ID.

```
tenant_id:  # The TenantID of the tenant in use
```

Kafka

Kafka secret

The Kafka secret is necessary for GCA to access Kafka. The Kafka secret is provisioned in the system

as `kafka-secrets` when you set up the environment for GCA. Configure the Kafka secret by creating a Helm chart override in the **values.yaml** file.

```
kafka:
  password:  # Credentials for accessing Kafka. This secret is created during deployment.
```

Kafka bootstrap

To allow the Kafka service on GCA to align with the infrastructure Kafka service, make a Helm override entry with the location of the Kafka bootstrap.

```
kafka:
  bootstrap:  # the Kafka address to align with the infrastructure Kafka
```

S3-compatible storage

If you are using S3-compatible object storage on GCP to store the GCA snapshot, modify the following `storage: s3` entries in the **values.yaml** file:

```
storage:
  ...
  s3:
    bucket:  # The bucket name
    gcaSnapshots:  # The volume or folder in the bucket where the GCA snapshot will be stored
    accessKey:  # The access key created when you created the bucket
    secretKey:  # The secret created when the bucket was provisioned
    endPoint:  # The bucket host
```

GKE example

```
storage:
  ...
  s3:
    bucket: "test-example-bucket-one"
    gcaSnapshots: "/gca"
    accessKey: ""
    secretKey: ""
    useSSL: true
    endPoint: "storage.googleapis.com"
    port: 443
    Insecure: true
```

Configuration database

Specify applicable details for the configuration database. The password you configure here is provisioned as `cfgdb-secrets` in the system.

```
cfgdb: # The applicable details for the Configuration Database, created before you deployed
the Tenant service
  name:  # The name of the database
  host:  # The host on which the DBMS is running
  username:  # The user account for GCA to access the database. The user account must have at
```

```
least read permissions
  password:  # The password for the user account
```

GIM database

Specify the applicable details for the Info Mart database. The password you specify here is provisioned in the system as `gimdb-secrets`.

```
gimdb: # The applicable details for the Info Mart database
  name:  # The name of the database
  host:  # The host on which the DBMS is running
  username:  # The user account created when you created the GIM database
  password:  # The password for the user account
```

Config Maps

There are no Config Maps for GCA you can configure directly.

Deploy GIM Config Adapter

Contents

Learn how to deploy GIM Config Adapter (GCA) into a private edition environment.

Related documentation:

-
-
-
-

RSS:

- For private edition

Assumptions

- The instructions on this page assume you are deploying the service in a service-specific namespace, named in accordance with the requirements on Creating namespaces. If you are using a single namespace for all private edition services, replace the namespace element in the commands on this page with the name of your single namespace or project.
- Similarly, the configuration and environment setup instructions assume you need to create namespace-specific (in other words, service-specific) secrets. If you are using a single namespace for all private edition services, you might not need to create separate secrets for each service, depending on your credentials management requirements. However, if you do create service-specific secrets in a single namespace, be sure to avoid naming conflicts.

> **Important**
>
> Make sure to review Before you begin GCA deployment for the full list of prerequisites required to deploy GCA, including provisioning the required S3-compatible storage.

Set up your environment

To prepare your environment for the deployment, complete the steps in this section for:

- GKE
- AKS

GKE environment setup

1. Ensure that the gcloud CLI and required Helm version are installed on the host where you will run the deployment.
2. Log in to the GKE cluster from the host where you will run the deployment:

   ```
   gcloud container clusters get-credentials
   ```

3. If the cluster administrator has not already done so, create a new namespace for GCA:
   - Create a .json file specifying the namespace metadata. For example, **create-gca-namespace.json**:

     ```
     {
       "apiVersion": "v1",
       "kind": "Namespace",
       "metadata": {
         "name": "gca",
         "labels": {
           "name": "gca"
         }
       }
     }
     ```

   - Execute the following command to create the namespace:

     ```
     kubectl apply -f apply create-gca-namespace.json
     ```

   - Confirm namespace creation:

     ```
     kubectl describe namespace gca
     ```

4. Create the pull secret for the image registry.
   - This step defines a secret so that Kubernetes can authenticate your image repository and pull artifacts from it. The repository is represented as `docker-registry` in the system. For information about downloading artifacts from the repository, see Downloading your Genesys Multicloud CX containers.
   - When you configure GCA, you will reference the registry pull secret as a Helm chart override; see GCA Helm chart overrides.

     ```
     kubectl create secret docker-registry  --docker-server= --docker-username= --docker-
      password= --docker-email= -n gca
     ```

AKS environment setup

1. Ensure that the Azure CLI and required Helm version are installed on the host where you will run the deployment.
2. Log in to the AKS cluster from the host where you will run the deployment.

   ```
   az aks get-credentials --resource-group  --name  --admin
   ```

3. If the cluster administrator has not already done so, create a new namespace for GCA:
   - Create a .json file specifying the namespace metadata. For example, **create-gca-namespace.json**:

     ```
     {
       "apiVersion": "v1",
     ```

```
  "kind": "Namespace",
  "metadata": {
    "name": "gca",
    "labels": {
      "name": "gca"
    }
  }
}
```

- Execute the following command to create the namespace:

  ```
  kubectl apply -f apply create-gca-namespace.json
  ```

- Confirm namespace creation:

  ```
  kubectl describe namespace gca
  ```

4. Create the pull secret for the image registry.

   - This step defines a secret so that Kubernetes can authenticate your image repository and pull artifacts from it. The repository is represented as `docker-registry` in the system. For information about downloading artifacts from the repository, see Downloading your Genesys Multicloud CX containers.
   - When you configure GCA, you will reference the registry pull secret as a Helm chart override; see GCA Helm chart overrides.

     ```
     kubectl create secret docker-registry  --docker-server= --docker-username= --docker-
      password= --docker-email= -n gca
     ```

Deploy

Execute the following command to install GCA:

```
helm upgrade --install  -f  -n gca
```

Execute the following command to install GCA monitoring:

```
helm upgrade --install gca-monitoring  -n gca
```

Validate the deployment

You can consider GCA deployment successful when the pod is running and in ready state. Genesys Info Mart does not report the ready state for pods until internal health checks are satisfied and the pods are operational. You can use standard `kubectl` commands like `list` and `get` to verify the successful deployment and readiness status of the Kubernetes objects, including connection to the database.

However, from a functional point of view, you cannot validate deployment of GCA unless GSP and GIM have been deployed as well. Do not expect consistent data until all three Genesys Info Mart services are up and running. For more details about functional checks you can perform to validate GCA deployment, see the equivalent validation section on the Deploy GIM Stream Processor page.

Upgrade, roll back, or uninstall Genesys Info Mart

Contents

Learn how to upgrade, roll back, or uninstall Genesys Info Mart.

Related documentation:

-
-
-
-

RSS:

- For private edition

> **Important**
>
> The instructions on this page assume you have deployed the services in service-specific namespaces. If you are using a single namespace for all private edition services, replace the namespace element in the commands on this page with the name of your single namespace or project.

Supported upgrade strategies

Genesys Info Mart supports the following upgrade strategies:

Service	Upgrade Strategy	Notes
Genesys Info Mart	Rolling Update	

For a conceptual overview of the upgrade strategies, refer to Upgrade strategies in the Setting up Genesys Multicloud CX Private Edition guide.

In general, upgrades for the three Genesys Info Mart services are released at the same time, but this is not necessarily the case. Note that the respective release packages have different release numbers.

As long as you remain within the range of N-2 releases for each service, the Genesys Info Mart services can inter-operate with one another. However, if upgraded releases for all the Genesys Info Mart services are available, Genesys recommends that you upgrade them all during the same upgrade cycle.

Upgrading GIM might or might not require migrating the Info Mart database as well, depending on whether there are schema changes. If database migration is required, the upgraded GIM service automatically launches the migration job. GIM upgrade and database migration do not interfere with aggregation or other queries in progress. Robust backward compatibility means that an upgraded or

rolled-back GIM service can operate with either a migrated or a nonmigrated Info Mart database, even with a half-migrated database if the migration job did not complete successfully.

Timing

A regular upgrade schedule is necessary to fit within the Genesys policy of supporting N-2 releases, but a particular release might warrant an earlier upgrade (for example, because of a critical security fix).

If the service you are upgrading requires a later version of any third-party services, upgrade the third-party service(s) before you upgrade the private edition service. For the latest supported versions of third-party services, see the Software requirements page in the suite-level guide.

Scheduling considerations

Genesys recommends that you upgrade the services methodically and sequentially: Complete the upgrade for one service and verify that it upgraded successfully before proceeding to upgrade the next service. If necessary, roll back the upgrade and verify successful rollback.

Do not attempt to upgrade a service unless all the Genesys Info Mart services are operating normally.

Upgrading the services potentially affects the availability of Info Mart data. Plan a time when this eventuality is least disruptive to your operations.

If you are upgrading all the services during the same upgrade cycle, upgrade GIM first, then GSP, then GCA.

Monitoring

Monitor the upgrade process using standard Kubernetes and Helm metrics, as well as service-specific metrics that can identify failure or successful completion of the upgrade (see Observability in Genesys Info Mart).

Genesys recommends that you create custom alerts for key indicators of failure — for example, an alert that a pod is in pending state for longer than a timeout suitable for your environment. Consider including an alert for the absence of metrics, which is a situation that can occur if the Docker image is not available. Note that Genesys does not provide support for custom alerts that you create in your environment.

Preparatory steps

Ensure that your processes have been set up to enable easy rollback in case an upgrade leads to compatibility or other issues.

Each time you upgrade a service:

1. Review the release note to identify changes.
2. Ensure that the new package is available for you to deploy in your environment.
3. Ensure that your existing **-values.yaml** file is available and update it if required to implement changes.
4. Review the Reporting and Analytics Aggregates (RAA) and Genesys CX Insights (GCXI) release notes and release advisories, for information about the possible impact of Genesys Info Mart migration on aggregation, as well as workarounds or additional steps to take during Genesys Info Mart migration.

In addition, if you are upgrading the GIM service:

1. Review the "New in the Info Mart Database" and "Summary of Info Mart Schema Changes" pages in the *Genesys Info Mart Physical Data Model*, to identify whether there are schema changes that will necessitate migration of the Info Mart database as part of a GIM upgrade. If so, and if you use the Data Export capability, decide whether you want to update your export views to include new data available in the migrated database.
2. If you do want to update your export views, do so after the GIM upgrade is complete (see Update export views).
3. For reference purposes, identify and make notes of any custom changes that you made to the Info Mart database—for example, table spaces, partitions, additional indexes, views, or permissions.
   Database migration might involve creating new tables or replacing some tables with views. You will need to re-create any custom database objects or permissions that become lost or invalidated during the update process.
4. Ensure that a recent backup of the Info Mart database is available in case you need to roll back a database migration.

Rolling Update

Rolling Update: Upgrade

Execute the following command to upgrade :

```
helm upgrade --install  -f -values.yaml  -n
```

Tip: If your review of Helm chart changes (see Preparatory Step 3) identifies that the only update you need to make to your existing **-values.yaml** file is to update the image version, you can pass the image tag as an argument by using the `--set` flag in the command:

```
helm upgrade --install  -f -values.yaml  --set .image.tag=
```

When you upgrade GIM and GSP, increase the Helm timeout value from the default 5 minutes to 15 minutes or longer. For GIM, you need a longer timeout to accommodate possible database migration, which can take a significant amount of time if, for example, new indexes are being added. For GSP, you need to allow sufficient time for GSP to run the process to create a pre-upgrade hook, to ensure that state information is correctly preserved when the previous instance shuts down.

Examples

- GIM and GIM Monitoring:

```
helm upgrade --install gim gim-100.0.116+2900.tgz -f gim-values.yaml -n gim --timeout 15m
 helm upgrade --install gim-monitoring gim-monitoring-100.0.116+2900.tgz -n gim
```

- GSP:

```
helm upgrade --install gsp gsp-100.0.100+1400.tgz -f gsp-values.yaml -n gsp --timeout 15m
```

- GCA and GCA Monitoring:

```
helm upgrade --install gca gca-100.0.100+2300.tgz -f gca-values.yaml -n gca
 helm upgrade --install gca-monitoring gca-monitoring-100.0.100+2300.tgz -n gca
```

Rolling Update: Verify the upgrade

Follow usual Kubernetes best practices to verify that the new service version is deployed. See the information about initial deployment for additional functional validation that the service has upgraded successfully.

Monitor the operations dashboard for the status of services and job execution. Genesys Info Mart does not report the Ready state for a pod until the pod is operational. The GIM service does not report the Ready state until the first post-upgrade transformation job has completed successfully.

To verify that an upgraded GIM is successfully populating the Info Mart database, use the **gim_kafka_timestamp_behind** metric to track data latency and validate that there is no growing backlog. To validate successful database migration, you can query the database directly to verify that schema changes (for example, new tables or columns) have been implemented and the new database objects are being populated.

If Genesys Info Mart is operating normally after an upgrade in which there were schema changes, but the migration job either did not run or else did not complete successfully, a transitory condition (for example, losing connection to the database or exceeding the Helm timeout for executing the job) was the likely cause of the migration job failure. Simply restart the GIM pod. The GIM service detects the out-of-date schema and restarts the migration job, which picks up from where it left off.

If a new pod does not start and/or operate successfully, Genesys recommends that you roll back to the previous version, so that Genesys Info Mart can continue operating while you investigate the reasons for failure. Rolling back the GIM service does not affect the Info Mart database.

If there are issues with a third-party dependency, such as Kafka or PostgreSQL, then rollback will not work until you resolve the third-party issue.

Contact Genesys Customer Care if you cannot resolve an upgrade issue.

Rolling Update: Rollback

Execute the following command to roll back the upgrade to the previous version:

```
helm rollback
```

or, to roll back to an even earlier version:

```
helm rollback
```

Alternatively, you can re-install the previous package:

1. Revert the image version in the `.image.tag` parameter in the **-values.yaml** file. If applicable, also revert any configuration changes you implemented for the new release.
2. Execute the following command to roll back the upgrade:

   ```
   helm upgrade --install  -f -values.yaml
   ```

   Tip: You can also directly pass the image tag as an argument by using the `--set` flag in the command:

   ```
   helm upgrade --install  -f -values.yaml  --set .image.tag=
   ```

When you roll back GIM and GSP, increase the Helm timeout value from the default 5 minutes to 15 minutes or longer.

Examples

- GIM:

  ```
  helm rollback gim --timeout 15m
  ```

- GSP:

  ```
  helm rollback gsp --timeout 15m
  ```

- GCA:

  ```
  helm rollback gca
  ```

Rolling Update: Verify the rollback

Verify the rollback in the same way that you verified the upgrade (see Rolling Update: Verify the upgrade).

Post-upgrade procedures

No routine post-upgrade procedures are required, but you might want to update your export views for Data Export if there were schema changes.

Update export views

If you use the Data Export feature, your data will continue to be exported using the export views in effect before GIM was upgraded, regardless of whether the upgrade was associated with schema changes that triggered migration of the Info Mart database.

If you want your exported data to reflect the schema changes (for example, new tables or columns), you must update your export views. Execute the following command, which forces GIM to rerun the migration job with the **make-export-views** parameter:

```
kubectl exec  -n  -- ./entrypoint.sh -job Job_MigrateGIM -make-export-views
```

You might also need to update your target schema and change your import processing to

accommodate the Info Mart schema changes. For more information, see About Data Export.

Uninstall

> ### Warning
> Uninstalling a service removes all Kubernetes resources associated with that service. Genesys recommends that you contact Genesys Customer Care before uninstalling any private edition services, particularly in a production environment, to ensure that you understand the implications and to prevent unintended consequences arising from, say, unrecognized dependencies or purged data.

Execute the following command to uninstall :

```
helm uninstall  -n
```

For example:

```
helm uninstall gsp -n gsp
```

Uninstalling the Genesys Info Mart services can be useful if, say, you have completed a beta testing program and want to clean up your environment and free up resources.

Uninstalling any of the Genesys Info Mart services does not affect the Info Mart database or any files on the S3-compatible storage location GIM uses for Data Export.

Uninstalling GIM does not affect aggregation that may be in process and does not directly affect GCXI reports. However, of course, the Info Mart database on which GCXI reporting is based will no longer be updated.

Uninstalling GSP does not remove all state-related Kubernetes resources (ConfigMaps) referring to files on the S3-compatible storage that GSP uses during processing. If you are uninstalling GSP preparatory to re-installing it, execute the following command to clear these resources:

```
kubectl delete cm --selector='app=’
```

where is the service name calculated by the **gsp.fullname macro**. If you did not override the default release name, then is gsp.

Observability in Genesys Info Mart

Contents

Learn about the logs, metrics, and alerts you should monitor for Genesys Info Mart.

Related documentation:

-
-
-
-

RSS:

- For private edition

Monitoring

Private edition services expose metrics that can be scraped by Prometheus, to support monitoring operations and alerting.

- As described on Monitoring overview and approach, you can use a tool like Grafana to create dashboards that query the Prometheus metrics to visualize operational status.
- As described on Customizing Alertmanager configuration, you can configure Alertmanager to send notifications to notification providers such as PagerDuty, to notify you when an alert is triggered because a metric has exceeded a defined threshold.

The services expose a number of Genesys-defined and third-party metrics. The metrics that are defined in third-party software used by private edition services are available for you to use as long as the third-party provider still supports them. For descriptions of available Genesys Info Mart metrics, see:

- GIM Config Adapter metrics
- GIM metrics
- GIM Stream Processor metrics

See also System metrics.

Enable monitoring

The Genesys Info Mart services use PodMonitor custom resource definitions (CRDs), which are defined in the Helm charts by default. No additional service-level configuration is required to enable monitoring. See Enabling monitoring for information about enabling monitoring for your private edition solution.

Service	CRD or annotations?	Port	Endpoint/ Selector	Metrics update interval
GIM Config Adapter	PodMonitor	9249	See selector details on the GIM Config Adapter metrics and alerts page	30 seconds
GIM	PodMonitor	8249	See selector details on the GIM metrics and alerts page	30 seconds
GIM Stream Processor	PodMonitor	9249	See selector details on the GIM Stream Processor metrics and alerts page	30 seconds

Configure metrics

The metrics that are exposed by the Genesys Info Mart services are available by default. No further configuration is required in order to define or expose these metrics. You cannot define your own custom metrics.

The Metrics pages linked to above show some of the metrics the Genesys Info Mart services expose. You can also query Prometheus directly or via a dashboard to see all the metrics available from the Genesys Info Mart services.

Alerting

Private edition services define a number of alerts based on Prometheus metrics thresholds.

> **Important**
> You can use general third-party functionality to create rules to trigger alerts based on metrics values you specify. Genesys does not provide support for custom alerts that you create in your environment.

For descriptions of available Genesys Info Mart alerts, see:

- GIM Config Adapter alerts
- GIM alerts
- GIM Stream Processor alerts

Configure alerts

Private edition services define a number of alerts by default (for Genesys Info Mart, see the pages linked to above). No further configuration is required.

The alerts are defined as **PrometheusRule** objects in a **prometheus-rule.yaml** file in the Helm charts. As described above, Genesys Info Mart does not support customizing the alerts or defining additional **PrometheusRule** objects to create alerts based on the service-provided metrics.

Logging

Genesys Info Mart uses a structured logging approach and outputs logs to standard output (stdout).

You can set the log levels for the Genesys Info Mart services in the respective values.yaml files. By default, the GSP and GIM logs are at the INFO level, while GCA logs are at DEBUG level.

See Logging overview and approaches and related links in the *Operations* guide for more information about logging in private edition, including deployment information and how you can use log collectors (such as Elasticsearch) to extract the logs and viewing tools (such as Kibana) to visualize the data.

GSP metrics and alerts

Contents

Find the metrics GSP exposes and the alerts defined for GSP.

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval

GSP	PodMonitor	9249	Endpoint: /
			Selector:
			`matchLabels:`
			`  app: {{ template "gsp.fullname" . }}`
			where the value of `gsp.fullname` depends on deployment parameters such as Helm release name, `.Values.fullnameOverride`, and `.Values.nameOverride`.
				30 seconds

See details about:

- GSP metrics
- GSP alerts

Metrics

GSP exposes some standard Apache Flink and Kafka metrics as well as Genesys-defined metrics, which are exposed via the Flink API. Therefore, all GSP metrics start with the prefix **flink_** but in some cases the values are calculated by GSP.

You can query Prometheus directly to see all the metrics Flink and the Flink Kafka connector expose through GSP.

- For full information about the standard Flink metrics, see the Apache Flink documentation.
- For full information about the Kafka metrics, see the Apache Kafka or Confluent Kafka documentation.

The following metrics are likely to be particularly useful. The naming convention is _. Genesys does not commit to maintain other currently available GSP metrics not documented on this page.

Metric and description	Metric details	Indicator of
flink_taskmanager_job_task_operator_errors_numInvalidRecords		Error

Metric and description	Metric details	Indicator of
Number of invalid input records.	**Type:** Gauge **Label:** **Sample value:** 0	
flink_jobmanager_numRunningJobs Number of running Flink jobs. If less than 1, there is a problem.	**Unit:** **Type:** Gauge **Label:** **Sample value:** 1	Error
flink_taskmanager_job_task_operator_user_errors_numOversizedMessages Number of messages exceeding the **max.request.size** Kafka option.	**Unit:** **Type:** Gauge **Label:** • operator_name **Sample value:** 0	Error
flink_taskmanager_job_task_operator_tenant_error_total Number of issues encountered, such as errors or warnings.	**Unit:** **Type:** Gauge **Label:** • operator_name • tenant • error **Sample value:**	Error
flink_taskmanager_job_task_operator_currentInputWatermark The last watermark received by this operator/task, in milliseconds since the Unix Epoch (00:00:00 UTC on 1 January 1970). **Note:** For operators/tasks with two inputs, this is the earlier of the last received watermarks.	**Unit:** milliseconds **Type:** Gauge **Label:** • operator_name **Sample value:**	Latency

flink_taskmanager_job_task_operator_currentOutputWatermark
The last watermark this operator has emitted, in milliseconds since the Unix Epoch.
	Unit: milliseconds
	Type: Gauge
	Label:
	• operator_name:
	  • Sink:_Agent_State_Facts
	  • Sink:_Interaction_Facts
	Sample value:
		Latency

flink_taskmanager_job_task_operator_records_lag_max The maximum lag in terms of the number of records for any partition in this window. An increasing value over time is your best indication that the consumer group is not	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Latency

Metric and description	Metric details	Indicator of
keeping up with the producers.		
flink_taskmanager_job_task_operator_records_consumed_rate The average number of records consumed per second.	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Traffic
flink_taskmanager_job_task_operator_numCallsCreated Total number of EventCallCreated events GSP received since it started processing.	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Traffic
flink_taskmanager_job_task_operator_numCallsCreatedPerSecond Number of EventCallCreated events per second (CPS).	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Traffic
flink_taskmanager_job_task_operator_numThreadsCreated Total number of CallThreads GSP received since it started processing.	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Traffic
flink_taskmanager_job_task_operator_numCallThreadsCreatedPerSecond Number of CallThreads per second (CTHPS).	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Traffic
flink_taskmanager_job_task_operator_numChainsProcessed Total number of EventOCSChainStartProcessing events GSP received since it started processing.	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Traffic
flink_taskmanager_job_task_operator_numChainsProcessedPerSecond Number of EventOCSChainStartProcessing events per second (CPS).	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Traffic
flink_(job|task)manager_Status_JVM_CPU_Load The recent CPU usage for the JVM process. The value is a double in the [0.0,1.0] interval, where a value of 0.0 means that none of the CPUs were running threads from the JVM process, while a value of 1.0 means that all CPUs were actively running threads from the JVM 100% of the time during the recent period being observed. A negative value means usage data is not available. For more information, see https://docs.oracle.com/javase/7/docs/jre/api/management/extension/com/sun/management/OperatingSystemMXBean.html#getProcessCpuLoad().	**Unit:** **Type:** Gauge **Label:** • pod **Sample value:**	Saturation
flink_(job|task)manager_Status_JVM_Memory_Direct_TotalCapacity	**Unit:** Bytes	Saturation

Metric and description	Metric details	Indicator of
The total capacity of all buffers in the direct buffer pool.	**Type:** Gauge **Label:** • pod **Sample value:**	
flink_(job|task)manager_Status_JVM_Memory_Direct_MemoryUsed The amount of memory used by the JVM for the direct buffer pool.	**Unit:** bytes **Type:** Gauge **Label:** • pod **Sample value:**	Saturation
flink_(job|task)manager_Status_JVM_Memory_NonHeap_Max The maximum amount of non-heap memory that can be used for memory management.	**Unit:** bytes **Type:** Gauge **Label:** • pod **Sample value:**	Saturation
flink_(job|task)manager_Status_JVM_Memory_NonHeap_Used The amount of non-heap memory currently used.	**Unit:** bytes **Type:** Gauge **Label:** • pod **Sample value:**	Saturation
flink_(job|task)manager_Status_JVM_Memory_Heap_Max The maximum amount of heap memory that can be used for memory management.	**Unit:** bytes **Type:** Gauge **Label:** • pod **Sample value:**	Saturation
flink_(job|task)manager_Status_JVM_Memory_Heap_Used The amount of heap memory currently used.	**Unit:** bytes **Type:** Gauge **Label:** • pod **Sample value:**	Saturation

Alerts

The alerts are based on Flink and Kubernetes cluster metrics.

The following alerts are defined for GSP.

Alert	Severity	Description	Based on	Threshold
GspFlinkJobDown	Critical	Triggered when the GSP Flink job is not running (number of running jobs equals to 0 or metric is not available)	flink_jobmanager_numRunningJobs	For 5 minutes
GspOOMKilled	Critical	Triggered when a GSP pod is restarted because of OOMKilled	kube_pod_container_status_restarts_total	0
GspNoTmRegistered	Critical	Triggered when there are no registered TaskManagers (or metric not available)	flink_jobmanager_numRegisteredTaskManagers	For 5 minutes
GspUnknownPerson	High	Triggered when GSP encounters unknown person(s)	flink_taskmanager_job_task_operator_tenant_error_tota	For 5 minutes

GCA metrics and alerts

Contents

Find the metrics GCA exposes and the alerts defined for GCA.

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval

GCA	PodMonitor	9249	`selector:`
			`  matchLabels:`
			`  app: {{ template "fullname" . }}`
			
			where the value of `fullname` depends on `.Values.tenant_id`.
				30 seconds

See details about:

- GCA metrics
- GCA alerts

Metrics

Use standard Kubernetes metrics to monitor the GCA service. For information about standard Kubernetes and other system metrics to monitor services, see System metrics.

Metric and description	Metric details	Indicator of

Alerts

Alerts are based on Kubernetes cluster metrics.

The following alerts are defined for GCA.

Alert	Severity	Description	Based on	Threshold
GcaPodCrashLooping	Critical	Triggered when a GCA pod is crash looping.	kube_pod_container_status_restarts_total	The restart rate is greater than 0 for 5 minutes

Alert	Severity	Description	Based on	Threshold
GcaOOMKilled	Critical	Triggered when a GCA pod is restarted because of OOMKilled.	kube_pod_container_status_restarts_total and kube_pod_container_status_last_terminated_reason	1

GIM metrics and alerts

Contents

Find the metrics GIM exposes and the alerts defined for GIM.

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval

GIM	PodMonitor	8249	Endpoint: /metrics
			Selector:
			`selector:`
			`  matchLabels:`
			`  app: {{ template "fullname" . }}`
			where the value of `fullname` depends on `.Values.tenant_id`.
				30 seconds

Metrics

The following metrics are defined for the GIM service.

Metric and description	Metric details	Indicator of
gim_server_state The state of the GIM server. Valid values are: • 0 — starting • 1 — running • -1 — failing once • -2 — failing twice • -3 — failing three or more times in a row	**Unit:** **Type:** Gauge **Label:** • applicationName • gim_version • built • schema_version **Sample value:**	Error
gim_failed_jobsteps Number of failed job steps.	**Unit:** **Type:** Gauge **Label:** **Sample value:**	Error
gim_number_failed_kafkajob	**Unit:**	Error

Metric and description	Metric details	Indicator of
Number of times the KAFKA step in the transform job failed.	**Type:** Gauge **Label:** • applicationName **Sample value:**	
gim_issues_total Number of encountered issues.	**Unit:** **Type:** Counter **Label:** • applicationName • issue **Sample value:**	Error
gim_kafka_timestamp_millis Maximum Kafka timestamp per topic per partition, in milliseconds since the Unix Epoch.	**Unit:** milliseconds **Type:** Gauge **Label:** **Sample value:**	Latency
gim_kafka_timestamp_behind Kafka latency per topic per partition, in milliseconds. Latency is calculated as (current system time) minus (maximum timestamp received). Therefore, this metric can grow if a topic or partition is idle and no records are received.	**Unit:** milliseconds **Type:** Gauge **Label:** • applicationName • topic • partition **Sample value:**	Latency

Alerts

No alerts are defined for Genesys Info Mart.