Genesys Info Mart Private Edition Guide

Deploy GIM Config Adapter

12/19/2025

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.