This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Outbound (CX Contact) Private Edition Guide

2/21/2026

Table of Contents

Overview

Configure and deploy

Upgrade, roll back, or uninstall

Observability

Contents

Find links to all the topics in this guide.

Related documentation:

-
-
-

RSS:

- For private edition

CX Contact is a service available with the Genesys Multicloud CX private edition offering.

Overview

Learn more about CX Contact, its architecture, and how to support high availability and disaster recovery.

- About CX Contact
- Architecture
- High availability and disaster recovery

Configure and deploy

Find out how to configure and deploy CX Contact.

- Before you begin
- Configure CX Contact
- Deploy CX Contact
- Provision CX Contact

Upgrade, roll back, or uninstall

Find out how to upgrade, roll back, or uninstall the CX Contact service.

- Upgrade, roll back, or uninstall CX Contact

Observability

Learn how to perform observability tasks in CX Contact.

- Monitoring
- Alerting
- Logging

About CX Contact

Contents

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

Related documentation:

-
-
-
-

RSS:

- For private edition

Welcome to the *Genesys Outbound (CX Contact) Private Edition Guide*. This document explains the provisioning, deployment, configuration, and start procedures for Outbound (CX Contact). The microservice that provides the outbound functionality is called Outbound (CX Contact) Service (CXCS). Because this guide covers the deployment of the service, CX Contact and CXCS terminology is used in much of the descriptive text and in any sample commands.

CX Contact is an omnichannel, outbound campaign management solution that enables you to proactively reach out to your customers in an agile and fully compliant way. It's designed to be easily managed by business users, providing the agility your organization needs when it comes to how and when to communicate with customers and prospects.

The CX Contact application provides a web UI, and contains a set of components that enable you to create, run, and manage outbound voice, SMS, and email campaigns. It is equipped with a built in self-service, email, and SMS content management system that enables easy and repeated use of pre-set campaign strategies. The list manager needs no database manipulation skills, and allows users to easily set profiles and segments to leverage different contact strategies and channels. Every uploaded contact record is enriched with global compliance data, enabling the business user to consistently manage all regulatory requirements in global, regional, or local level.

Supported Kubernetes platforms

CX Contact is supported on the following platforms:

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

See CX Contact Release notes to see when the support was introduced.

Architecture

Contents

Learn about CX Contact architecture

Related documentation:

-
-
-
-

RSS:

- For private edition

Introduction

CX Contact is set of microservices that run in Kubernetes containers, each scalable in N+1 horizontal mode. It has a state-of-the-art user interface (UI) and middleware components, and uses Genesys servers on the back end (Configuration Server, Outbound Contact Server (OCS), and Stat Server). Genesys Web Services (GWS) is a prerequisite.

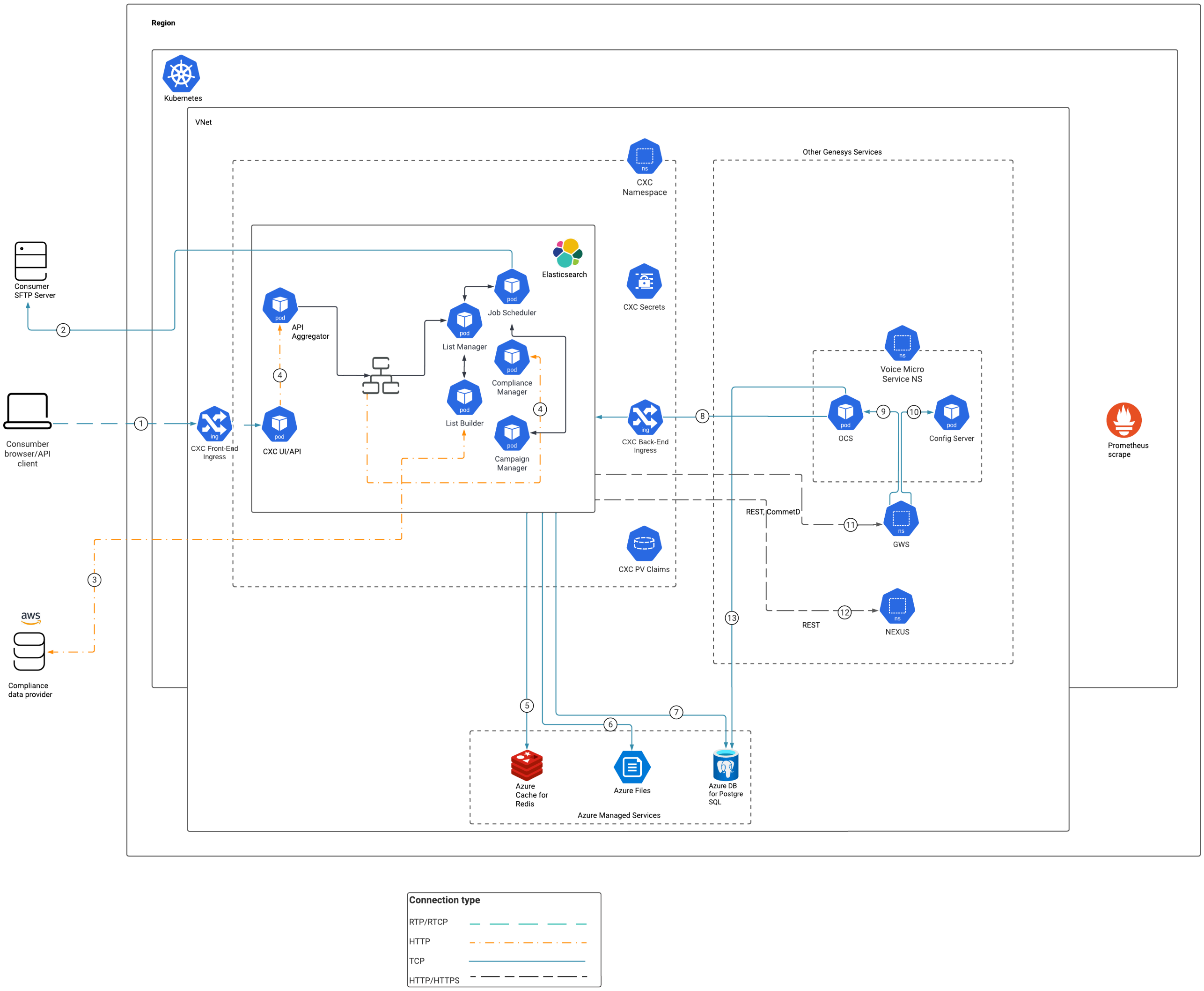
CX Contact supports Horizontal Pod Autoscaler (HPA) for Compliance Manager and Dial Manager.

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

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

Architecture diagram — Connections

The numbers on the connection lines refer to the connection numbers in the table that follows the diagram. 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 CX Contact as a service in the network.



Connections table

The connection numbers refer to the numbers on the connection lines in the diagram. 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 CX Contact as a service in the network. *Egress* means the CX Contact service is the source, and *Ingress* means the CX Contact 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	Customer Browser/API	CXC Front-end Ingress	RTP/RTCP	443	Ingress	RTP connection to CX Contact from a customer browser or API client.
2	CX Contact Job Scheduler	SFTP server	TCP	20, 21, 22	Egress	A connection for remote authentication on customer-specified SFTP server.
3	Compliance data provider	CX Contact List Builder	HTTP	443	Egress	A connection to read compliance data and rules from a compliance data provider.
4	CX Contact UI	CX Contact Backend	HTTP	3004-3008	Intra-cluster	HTTP connection between the CX Contact UI and backend services.
5	CX Contact	Redis	TCP	6379	Egress	TCP connection between CX Contact and Redis for caching data on user sessions and dialing history.
6	CX Contact	Network file share	TCP	2049	Egress	A connection for network/ cloud file storage to store import and export files for both contacts and suppression lists.
7	CX Contact	PostgreSQL	TCP	5432	Egress	Connection

Connection	Source	Destination	Protocol	Port	Classification	Data that travels on this connection
						between CX Contact and PostgreSQL to store data about contact and suppression lists from different outbound campaigns.
8	Outbound	CX Contact	TCP	8888	Ingress	Connection between Outbound contact server (back-end components) and CX Contact.
9	GWS	Outbound	TCP	5050	Intra-cluster	A connection between GWS and Outbound contact server for config data.
10	GWS	Config server	TCP	8888	Intra-cluster	A connection between GWS and Config server for config data.
11	CX Contact	GWS	HTTP/HTTPS	443	Egress	An HTTPS connection between CX Contact and GWS for managing config data, campaigns, tenant settings, and authentication.
12	CX Contact	Nexus	HTTP/HTTPS	443	Egress	An HTTPS connection between CX Contact and Nexus for SMS/Email

Connection	Source	Destination	Protocol	Port	Classification	Data that travels on this connection
						support.
13	Outbound	PostgreSQL	TCP	5432	Egress	A connection for Outbound DB access to update contact list data.

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?
CX Contact	N = N (N+1)	Not supported	Primary or secondary unit

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

CX Contact does not support Disaster Recovery or any kind of cross-regional deployment.

In most scenarios, CX Contact is deployed in the primary region. If deployed in supplementary regions, each deployment is completely independent from the other, and pods in different regions do not communicate with each other.

Before you begin

Contents

Find out what to do before deploying CX Contact.

Related documentation:

-
-
-
-

RSS:

- For private edition

Limitations and assumptions

There are no limitations. Before you begin deploying the CX Contact service, it is assumed that the following prerequisites and optional task, if needed, are completed:

Prerequisites

- A Kubernetes cluster is ready for deployment of CX Contact.
- The Kubectl and Helm command line tools are on your computer.
- You have connectivity to target cluster, the proper kubectl context to work with the cluster, and your user has administrative permissions to deploy CX Contact to the defined namespace.

Optional tasks

- **SFTP Server**—Install an SFTP Server with basic authentication for optional input and output data. SFTP Server is used when automation capabilities are required.
- **CDP NG access credentials**—As of CX Contact 9.0.025, Compliance Data Provider Next Generation (CDP NG) is used as a CDP by default. Before attempting to connect to CDP NG, obtain the necessary access credentials (ID and Secret) from Genesys Customer Care.
- **Bitnami repository**—If you choose to deploy dedicated Redis and Elasticsearch for CX Contact, add the Bitnami repository to install Redis and Elasticsearch using the following command:
  ```
  helm repo add bitnami https://charts.bitnami.com/bitnami
  ```

After you've completed the mandatory tasks, check the Third-party prerequisites.

Download the Helm charts

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

See Helm charts and containers for CX Contact for the Helm chart version you must download for your release.

CX Contact is the only service that runs in the CX Contact Docker container. The Helm charts included with the CX Contact release provision CX Contact and any Kubernetes infrastructure necessary for CX Contact to run.

Third-party prerequisites

Set up Elasticsearch and Redis services as standalone services or installed in a single Kubernetes cluster.

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

Third-party services

Name	Version	Purpose	Notes
Elasticsearch	7.x	Used for text searching and indexing. Deployed per service that needs Elasticsearch during runtime.	CX Contact supports Elasticsearch 6.3 and later releases.
Redis	6.x	Used for caching. Only distributions of Redis that support Redis cluster mode are supported, however, some services may not support cluster mode.	CX Contact supports Redis 4.0 (5.0 and later releases recommended), clustered with persistence in Production.
Load balancer		VPC ingress. For NGINX Ingress Controller, a single regional Google external network LB with a static IP and wildcard DNS entry will pass HTTPS traffic to NGINX Ingress Controller which will terminate SSL traffic and will be setup as part of the platform setup.	
PostgreSQL	11.x	Relational database.	Relational database.
A container image registry and Helm chart		Used for downloading Genesys containers and	

Name	Version	Purpose	Notes
repository		Helm charts into the customer's repository to support a CI/CD pipeline. You can use any Docker OCI compliant registry.	

Storage requirements

CX Contact requires shared persistent storage and an associated storage class created by the cluster administrator. The Helm chart creates the ReadWriteMany (RWX) Persistent Volume Claim (PVC) that is used to store and share data with multiple CX Contact components.

The minimal recommended PVC size is 100GB.

Network requirements

This topic describes network requirements and recommendations for CX Contact in private edition deployments:

Single namespace

Deploy CX Contact in a single namespace to prevent ingress/egress traffic from going through additional hops, due to firewalls, load balancers, or other network layers that introduce network latencies and overhead. Do not hardcode the namespace. You can override it by using the Helm file/values (provided during the Helm install command **standard --namespace= argument**), if necessary.

External connections

For information about external connections from the Kubernetes cluster to other systems, see Architecture. External connections also include:

- Compliance Data Provider (AWS)
- SFTP Servers

Ingress

The CX Contact UI requires Session Stickiness. Use **ingress-nginx** as the ingress controller (see github.com).

Important

The CX Contact Helm chart contains default annotations for session stickiness only for **ingress-nginx**. If you are using a different ingress controller, refer to its documentation for session stickiness configuration.

Ingress SSL

If you are using Chrome 80 or later, the **SameSite** cookie must have the **Secure** flag (see Chromium Blog). Therefore, Genesys recommends that you configure a valid SSL certificate on ingress.

Logging

Log rotation is required so that logs do not consume all of the available storage on the node.

Kubernetes is currently not responsible for rotating logs. Log rotation can be handled by the **docker json-file log driver** by setting the **max-file** and **max-size** options.

For effective troubleshooting, the engineering team should provide **stdout logs** of the pods (using the command **kubectl logs**). As a result, log retention is not very aggressive (see JSON file logging driver). For example:

```
{
  “log-driver”: “json-file”,
  “log-opts”: {
    “max-size”: “100m”,
   “max-file”: “3”
  }
}
```

For on-site debugging purposes, CX Contact logs can be collected and stored in Elasticsearch. (For example, EFK stack. See medium.com).

Monitoring

CX Contact provides metrics that can be consumed by Prometheus and Grafana. It is recommended to have the **Prometheus Operator** (see githum.com) installed in the cluster. CX Contact Helm chart supports the creation of **CustomResourceDefinitions** that can be consumed by the Prometheus Operator.

For more information about monitoring, see Observability in Outbound (CX Contact).

Browser requirements

Browsers

Name	Version	Notes
Chrome	Current release or one version	Chrome updates itself

Name	Version	Notes
	previous	automatically. Versions of Chrome are only an issue if your IT department restricts automatic updates.The latest version of Chrome must be used as the CX Contact UI browser.
Microsoft Edge (Legacy)	Current release	Starting from CX Contact release 9.0.026.04, Edge Chromium browser 2020

Genesys dependencies

CX Contact components operate with Genesys core services (v8.5 or v8.1) in the back end. All voice-processing components (Voice Microservice and shared services, such as GVP), and the GWS and Genesys Authentication services (mentioned below) must deployed and running before deploying the CX Contact service. See Order of services deployment.

The following Genesys services and components are required:

- GWS
- Genesys Authentication Service
- Tenant Service
- Voice Microservice
- Multi-tenant Configuration Server

Nexus is optional.

GDPR support

CX Contact does not support GDPR.

Configure CX Contact

Contents

Learn how to configure CX Contact.

Related documentation:

-
-
-
-

RSS:

- For private edition

Override Helm chart values

You can specify parameters for the deployment by overriding Helm chart values in the **values.yaml** file. See the Parameters table for a full list of overridable values.

For more information about Helm chart values, see Overriding Helm chart values.

Parameter	Description
configserver.user_name, user_password	Defines the system username and password for CX Contact.
redis.nodes	Provides a valid URI to Redis.
redis.password	Provides a valid auth password for Redis.
elasticsearch.host	Provides a valid URI to Elasticsearch.
gws.client_id	The name of the GWS service client that will be created (if it doesn't exist) and the secret that will be placed in the k8s secrets repository.
gws.client_secret	The client that will be created with this secret string. If a GWS client with this name already exists, you'll need to enter the secret here.
gws.frontend_host, frontend_port	The SSO GAuth URI where CX Contact redirects during log in.
core.auth, environment	The internal URI to core services that is required for further provisioning. You can see, in our example GAuth is installed in namespace "gauth"
platform.ocs, configration, .. etc.	The internal URI to the platform's GWS services. You can see, in our example GWS is installed in namespace "gws"
ingress.cxc_frontend	Creates a URI that is used by Ingress to route external incoming requests to CX Contact (Web UI and API).

Parameter	Description
internal_ingress.cxc_backend	Creates the URI that is used by Ingress to route internal incoming requests to CX Contact (API for OCS, GWS, Designer, etc)
storage.size	Defines the appropriate size for the permanent storage, depending on the daily volume of interactions, etc.
storage.storageClassName	Picks the existing Storage Class, which is described in this document earlier.

Configure Kubernetes

Preconfiguring Kubernetes ConfigMaps and create a default secret when you are preparing the cluster resources.

Configure security

When configuring CX Contact, you must set the connectivity to the Compliance Data Provider (CDP).

> ### Tip
> Before attempting to connect to CDP Next Generation (NG), you'll need the access ID and Secret. To obtain these credentials, contact Genesys Customer Care.

As of 9.0.025.xx, CX Contact uses CDP NG by default. The following Helm chart settings control the CDP NG connectivity:

```
cxcontact:
  compliance_data:
    cdp_ng:
      url: "https://api.usw2.pure.cloud/api/v2/outbound/compliancedata"
      gcloud_auth: "https://login.usw2.pure.cloud/oauth/token"
      gcloud_id:
      gcloud_secret:
    # LIST_BUILDER_DATA_EMBEDDED_BASEPATH
    embedded_basepath: "/list_builder/data/ng_init_data"
    rule_set:
      areacode: "AU,CA,GB,NZ,US"
      geo: "AU,CA,GB,NZ,US"
      postal: "CA,GB,US"
      dnc: "GB,US"
```

> ### Important

> The **gcloud_id** and **gcloud_secret** parameters are required, but do not have default values.

You can use the following parameters to switch to legacy CDP:

```
cxcontact:
  compliance_data:
    cdp_ng:
      url: false
      gcloud_auth: false
      gcloud_id: false
      gcloud_secret: false
    # LIST_BUILDER_DATA_EMBEDDED_BASEPATH
    embedded_basepath: "/list_builder/data/init_data"
```

Security Context

The security context settings define the privilege and access control settings for pods and containers.

By default, the user and group IDs are set in the **values.yaml** file as `500:500:500`, meaning the **genesys** user. For example:

```
securityContext:
    runAsNonRoot: true
    runAsUser: 500
    runAsGroup: 500
    fsGroup: 500
```

TLS authentication

TLS 1.2 connectivity is required for all connections to databases (Redis, PostgreSQL, and Elasticsearch) and connections must be authenticated using credentials.

Deploy CX Contact

Contents

Learn how to deploy CX Contact 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**
>
> Review Before you begin for the full list of prerequisites required to deploy CX Contact.

CX Contact is a shared service and is deployed in each region, as required. After deployment, it will be fully functional only in the tenant's primary region.

Prepare cluster resources

To prepare your cluster resources, create a storage class.

> Tip
>
> Creating the storage class is optional. If you have an existing storage class, you can use it and a different provisioner, but the storage class must have ReadWriteMany (RWX) capabilities.

Create the Storage Class

1. Log in to the cluster using the Command Line Interface (CLI).
2. Go to **Storage**, and click **Storage Claim** > **Create Storage Class**.
3. Click **Edit YAML**.
4. Update the values in the template using the example below that works with the setup of your environment.
    - If your cluster is on-premises (NFS-based storage):

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: cxc-storage
      provisioner: cluster.local/nfs-vce-c00ds-vol1-nfs-subdir-external-provisioner
      reclaimPolicy: Delete
      volumeBindingMode: Immediate
      ```

    - If your cluster is in the cloud (Azure files-based storage):

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: cxc-storage
      provisioner: kubernetes.io/azure-file
      reclaimPolicy: Delete
      volumeBindingMode: Immediate
      ```

5. Click **Create**.

Deploy CX Contact

Complete the following procedure using Kubectl and the Helm tools:

1. Create a new project or a namespace.
    - For Kubernetes, enter

      ```
      kubectl create namespace cxc
      ```

    - For GKE, enter

      ```
      kubectl create ns cxc
      ```

    - For AKS, enter

```
kubectl create namespace cxc
```

2. Create the pull secret.Use the following code snippets as examples of how to create the default pull secret:
   - For generic Kubernetes:

     ```
     oc create secret docker-registry mycred --docker-server= --docker-username= --docker-password=
     ```

   - For GKE :

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

   - For AKS:

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

3. Download latest version of the CX Contact installation Helm Charts from the artifactory. See the JFrog Platform Artifactory.
4. Extract parameters from chart to see multiple (default) values used to fine-tune the installation.

   ```
   helm install cxc ./cxcontact-.tgz > values.yaml
   ```

   You can apply multiple override values to customize your setup. However, Genesys recommends using minimal overriding values in the installation:
   For example, override_values.yaml

   ```
   configserver:
     user_name: cloudcon
     user_password: cloudcon

    cxcontact:

     replicas: 2
     log:
       level: info

     compliance_data:
       cdp_url: false
       cdp_ng:
         gcloud_id: false
         gcloud_secret: false

    #  override:               #if connecting to Nexus. Otherwise Dial Manager is off
    #    dial-manager:
    #      enabled: false
    #      nexus:
    #        host: ..
    #        port: ..
    #      api_key: ..          #required

    monitoring:
     enabled: true
     dashboards: false
     alarms: false
     pagerduty: false


    redis:
   ```

```
  enabled: true
  cluster: true
  # can be comma-delimited list of redis nodes, for e.g.
  # nodes: redis://redis-node1:6379,redis://redis-node2:6379,redis://redis-node3:6379
  nodes: redis://infra-redis-redis-cluster.infra.svc.cluster.local:6379
  #use_tls: false
  requirepass: true
  password:

elasticsearch:
  enabled: true
  host: http://elastic-es-http.infra.svc.cluster.local
  port: 9200

gws:
  # secret in plain text
  client_id: cx_contact
  client_secret: cx_contact

  # GWS Ingress URL
  frontend_host: https://gauth.apps.
  frontend_port: 443


#  Services. Will be used for connection to GWS if GWS Internal Ingress URL is disabled
(empty values)
  core:
    auth:
      host: http://gauth-auth.gauth
      port: 80
    environment:
      host: http://gauth-environment.gauth
      port: 80
  platform:
    ocs:
      host: http://gws-service-proxy.gws
      port: 80
    configuration:
      host: http://gws-service-proxy.gws
      port: 80
    statistics:
      host: http://gws-service-proxy.gws
      port: 80
    setting:
      host: http://gws-service-proxy.gws
      port: 80
    voice:
      host: http://gws-service-proxy.gws
      port: 80


ingress:
  enabled: true
  #tls_enabled: false
  cxc_frontend: cxc.apps.
  annotations:
#    !!!Ingress Session Stickiness is required.
#    Default annotations:
    nginx.ingress.kubernetes.io/affinity: cookie
    nginx.ingress.kubernetes.io/session-cookie-samesite: "Lax"
    nginx.ingress.kubernetes.io/session-cookie-name: "cxc-session-cookie"
    nginx.ingress.kubernetes.io/proxy-body-size: "0"
  tls: []
  #  - hosts:
  #      - chart-example.local
  #    secretName: chart-example-tls

# Additional ingress to expose internal backend endpoints.
# If disabled  - all endpoints will be exposed on ingress.cxc_frontend
internal_ingress:
  enabled: true
  tls_enabled: false
  cxc_backend: cxc-int.apps.
  annotations:
#    Default annotations:
    nginx.ingress.kubernetes.io/proxy-body-size: "0"
    nginx.ingress.kubernetes.io/ssl-redirect: 'false'
  tls: []
  #  - hosts:
  #      - chart-example.local
  #    secretName: chart-example-tls

storage:
  # Persistent Volumes Claim Configuration
  pvc:
    enabled: true
    create: true
#    Instructs Helm to skip deleting PVC when a helm operation (such as helm uninstall,
helm upgrade or helm rollback)
#    would result in its deletion. However, this resource becomes orphaned. Helm will no
longer manage it in any way.
#    https://helm.sh/docs/howto/charts_tips_and_tricks/#tell-helm-not-to-uninstall-a-
resource
#    If PVC is already orphaned and you want to re-use it - set `storage.pvc.create` to
`false`.
    keepAfterDeletion: false
    size: 10Gi
    name: cxc-claim
    storageClassName: cxc-storage
```

5. Validate the Helm chart and provided values, enter:

   ```
   $ helm template cxc ./cxcontact-.tgz -f override_values.yaml
   ```

6. Install the CX Contact chart, using the override values file that you prepared in step, enter:

   ```
   $ helm install cxc ./cxcontact-.tgz -f override_values.yaml
   ```

7. If errors occur, verify the input values, YAML files syntax, and your Kubernetes context. enter:

   ```
   $ kubectl config get-contexts
    $ kubectl get pods -n cxc
    $ kubectl describe pod  -n cxc
   ```

8. If troubleshooting is necessary, try adding the **--dry-run** command line parameter in **helm install ..** for verbose error output.

Tip

- To see the full set of available parameters, extract the default helm values from helm package:

> ```
> $ helm show values cxcontact-.tgz > values.yaml
> ```
>
> - For persistent volume claims in production, Genesys recommends 100-200 GB.

This completes the CX Contact shared service installation.

Next steps:

- Provision tenant for CX Contact/outbound. See Provision CX Contact.
- Validate the deployment.

Validate the deployment

1. Watch the helm output at the end of installation. It provides the status and additional information about where to log in to the CX Contact UI. See the following sample output:

   ```
   Release "cxc" has been upgraded. Happy Helming!
    NAME: cxc
    LAST DEPLOYED: Tue Jul 13 10:18:07 2021
    NAMESPACE: cxc
    STATUS: deployed
    REVISION: 1
    TEST SUITE: None
    NOTES:
    Please be patient while CXContact  is being deployed


    ENDPOINTS:
    UI: http://cxc.apps./ui/cxcontact/#!/campaign/list
    API basepath: http://cxc.apps./cx-contact/v3/
    API swagger doc: http://cxc.apps./cx-contact/v3/explore
    CXC backend basepath: http://cxc-int.apps./
   ```

2. Check the status of the Events and Pods. Pods should be up and running—8 CX Contact components in total (7, if you have not defined Nexus for digital outbound).
3. Check the **amark-app** and other pods logs, primarily for errors related to connectivity to Redis and Elasticsearch.
4. If you have provisioned tenants for CX Contact, log in to the CX Contact UI (use URL from the Helm output above) using the tenant's administrator credentials. If you can log in successfully, that confirms that CX Contact works with the Redis cluster:
   - Check the CX Contact **About** > **Versions** page for the health status of the CX Contact components. They should be all green.
   - Check the **Analytics** page. It should show your successful log in, and confirm that CX Contact works with Elasticsearch.
   - Try to create a test Contact list. If you succeed, CX Contact will display a success confirmation

message.

Configure monitoring and logging

CX Contact monitoring is enabled by default.

See monitoring details and how to configure logging parameters in Observability in Outbound (CX Contact).

Provision CX Contact

Contents

- Administrator

Learn how to provision CX Contact.

Related documentation:

-
-
-
-

RSS:

- For private edition

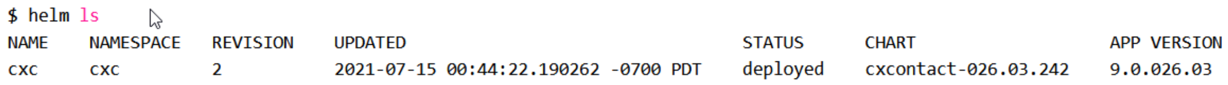
Prerequisites

Before you begin to provision tenants in CX Contact, ensure the following prerequisites are met:

- Ensure CX Contact is deployed. See Deploy CX Contact.
- Ensure the tenant exists in the GWS environment. For example, on the local machine, enter:
  `$ curl -u https:///environment/v3/environments`
  Here's a sample output:

```
{
    "status": {
        "code": 0
    },
    "data": {
        "genesysEnvironments": [
            {
                "id": "9350e2fc-a1dd-4c65-8d40-1f75a2e080dd",
                "tenant": "Environment",
                "appName": "Cloud",
                "username": "default",
                "password": "password",
                "connectionProtocol": "addp",
                "localTimeout": 5,
                "remoteTimeout": 7,
                "traceMode": "CFGTMBoth",
                "tlsEnabled": false,
                "configServers": [
                    {
                        "primaryAddress": "tenant-9350e2fc-
a1dd-4c65-8d40-1f75a2e080dd.voice.svc.cluster.local",
                        "primaryPort": 8888,
                        "readOnly": false,
                        "locations": "/USW1",
                        "readFromDb": false,
                        "useConfigExporter": false,
                        "initDb": false
                    }
                ],
                "proxyPort": 0
            }
        ]
    }
}
```

- Ensure the "cxc" Helm release is deployed (during installation of CX Contact). For example:



```
$ helm ls
NAME    NAMESPACE   REVISION    UPDATED                                 STATUS      CHART                   APP VERSION
cxc     cxc         2           2021-07-15 00:44:22.190262 -0700 PDT    deployed    cxcontact-026.03.242    9.0.026.03
```

Tenant provisioning

To provision tenants, you'll use the same Helm Charts as you used when deploying CX Contact, adding one additional overriding values YAML file. You should still use the same base override values file (**override_values.yaml**) that you used when deploying CX Contact.

> ### Important
> The **primary_host** parameter represents the primary Configuration Server's domain name within the cluster. It's important that this parameter is configured correctly, as it must match the **configServers.primaryAddress** parameter in the GWS environment (see Prerequisites) to ensure the Helm Chart uses the existing environment and environment ID.

1. Prepare the `provisioning_values.yaml` file, as follows:

```
# CXContact Tenant Provisioning configuration
tenant_provisioning:
  enabled: true
  # Basic Authentication for GWS Services. Required if `create_auth_client: true` or
`create_environment: true`.
  # Should be plain text
  gws_basic_auth_user: ops
  gws_basic_auth_pass: ops
  # Tenants list, that should be configured by CXC Tenant Provisioning. May contain
multiple tenants
  tenants:
    # Tenant 0
    - configserver:
        # if set to 'true' - will create environment if it doesn't exist. Else will re-
use existing.
        # if set to 'false' - will NOT create environment if it doesn't exist. Will use
existing.
        create_environment: true
        # should be unique
        primary_host:
```

```
        primary_port: 8888
        backup_host:
        backup_port:
        # Username and Password that will be used for creation of environment. Should
exist.
        username: default
        password: password
        # Configserver location e.g /USW1
        # corresponds to gws_configuration ENV GWS_CONFIGURATION_COMMON_LOCATION
        location: /USW1
        # GWS Server application name. Standard name is `CloudCluster`
        server_app_name: CloudCluster
        # GWS Client application name for GWS Connection. Standard name is `Cloud`
        client_app_name: Cloud
        # Outbound Contact Server Application Name
        ocs_name: OCS
        # Database Access Point Application Name
        ocs_dap_name: OCSDAP
        # CXContact requires set of options to be configured for OCS and CloudCluster
applications.
        # Will not update app options if set to false.
        update_app_options: true
      # The short tenant name (for example 22-06), should be unique
      short_tenant_name: ten100
      # The customer name (for example cxc), should be unique
      customer_name: Tenant100
      # Domain, will be used for login, should be unique
      domain: t100
```

2. Validate the Helm Chart and values. Enter:
   $ helm template cxc ./cxcontact-.tgz -f override_values.yaml -f provisioning_values.yaml
3. Upgrade the existing CX Contact Helm deployment with provisioning using the values file that you've just prepared. Enter:
   $ helm upgrade cxc ./cxcontact-.tgz -f override_values.yaml -f provisioning_values.yaml
4. If you encounter errors, verify the input values, YAML files syntax, and your Kubernetes context.

Tip

As long as there are no changes to the override values, you can rerun the provisioning multiple times for the same tenant. It will not affect the CX Contact deployment or corrupt tenant's configuration.

Validate tenant provisioning

At the end of the installation, be sure to check the Helm Chart output. It will provide the status and other information about where to log in to the CX Contact UI. In addition to the standard CX Contact installation output you will see the following provisioning information:

```
Following tenants were provisioned:

0) tenant-9350e2fc-a1dd-4c65-8d40-1f75a2e080dd.voice.svc.cluster.local
Domain for login to this tenant: t100
```

```
Test Username: t100\cxc_genesys@Tenant100.com
Provisioning logs can be accessed via `kubectl logs` command:
    $ kubectl -n cxc logs -f -l job-name=cxc-provisioning-0 -c cxc-provi
sioning --tail 9999

* Password is configured in helm cxc overrides, see variable configserver.user_password
```

Recommendations

Finally, note the following recommendations:

- Log in to the CX Contact UI using the URL from the Helm Chart output above and the provisioned tenant's Administrator credentials.
- Check the CX Contact **About** > **Versions** page, which contains the health statuses of the CX Contact components. They should all be green.
- Check CX Contact **Analytics** page, which should show your successful log in.
- Try to create a test Contact List. If you do it right, CX Contact will display a confirmation message that you were successful.

Upgrade, roll back, or uninstall CX Contact

Contents

Learn how to upgrade, roll back, or uninstall CX Contact.

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

Outbound (CX Contact) supports the following upgrade strategies:

Service	Upgrade Strategy	Notes
CX Contact	Rolling Update	

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

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.

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 Outbound (CX Contact)).

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.

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=
```

Example command:

```
helm upgrade cxc cxcontact-.tgz -f overrides.yaml
```

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.

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=
    ```

Example command:

```
helm rollback cxc
```

Rolling Update: Verify the rollback

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

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
```

Example command:

```
helm uninstall cxc
```

Observability in Outbound (CX Contact)

Contents

Learn about the logs, metrics, and alerts you should monitor for Outbound (CX Contact).

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 Outbound (CX Contact) metrics, see:

- CX Contact API Aggregator metrics
- CX Contact Campaign Manager metrics
- CX Contact Compliance Manager metrics
- CX Contact Dial Manager metrics
- CX Contact Job Scheduler metrics
- CX Contact List Builder metrics
- CX Contact List Manager metrics

See also System metrics.

Enable monitoring

CX Contact monitoring is enabled by default.

The following Kubernetes objects are created, based on default parameter settings in the Helm chart:

- **ServiceMonitor**—Prometheus operator uses this object to auto-discover endpoints for metrics scraping.
- **PrometheusRule**— AlertManager uses this object to import alert rules.
- Several **ConfigMaps**—Grafana uses these objects to import dashboards.

Service	CRD or annotations?	Port	Endpoint/ Selector	Metrics update interval
CX Contact API Aggregator	ServiceMonitor	9102	/metrics	15 seconds
CX Contact Campaign Manager	ServiceMonitor	3106	/metrics	15 seconds
CX Contact Compliance Manager	ServiceMonitor	3107	/metrics	15 seconds
CX Contact Dial Manager	ServiceMonitor	3109	/metrics	15 seconds
CX Contact Job Scheduler	ServiceMonitor	3108	/metrics	15 seconds
CX Contact List Builder	ServiceMonitor	3104	/metrics	15 seconds
CX Contact List Manager	ServiceMonitor	3105	/metrics	15 seconds

Configure metrics

The metrics that are exposed by the CX Contact 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 the metrics the CX Contact services expose. You can also query Prometheus directly or via a dashboard to see all the metrics available from the CX Contact services.

Alerting

No alerts are defined for Outbound (CX Contact).

Logging

Setting the logging parameters

Set/override the logging-related parameters to change the default Helm chart values as follows, if required:

```
cxcontact:
  log:
    level: info
    # logs can be saved to .log files on log volume.
    log_to_file: false
    # Log rotation.
    # false - will store logs indefinitely, needs rotation configured on the volume level.
    # true - will keep last 10 files with size up to 100mb
    log_rotation: true
    # Log volume configuration, If using persistentVolumeClaim - it should be created outside
of helm chart.
    log_volume_config:
      hostPath:
        # path on k8s nodes, that will be mounted to the pods
        # IMPORTANT! Should allow Write access to user with uid:guid 500:500!
        path: /mnt/log/cxc
```

CX Contact logs to stdout, by default. You can override this setting by changing the following parameter value to `true`:

```
log_to_file=true
```

API Aggregator metrics and alerts

Contents

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

Related documentation:

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
API Aggregator	ServiceMonitor	9102	/metrics	15 seconds

See details about:

- API Aggregator metrics
- API Aggregator alerts

Metrics

Here are some of the metrics exposed by API aggregator.

Metric and description	Metric details	Indicator of
cxc_api_aggregator_schedules_created_total Total schedules created.	**Unit:** **Type:** Counter **Label:** **Sample value:** 42	
cxc_api_aggregator_schedules_removed_total Total schedules removed.	**Unit:** **Type:** Counter **Label:** **Sample value:** 42	
cxc_api_aggregator_campaign_template_created_total Total campaign templates created.	**Unit:** **Type:** Counter **Label:** **Sample value:** 42	
cxc_api_aggregator_campaign_template_removed_total Total campaign templates removed.	**Unit:** **Type:** Counter **Label:** **Sample value:** 42	
cxc_api_aggregator_users_logged_in_total Total number of users who are logged in.	**Unit:** **Type:** Gauge	

Metric and description	Metric details	Indicator of
	Label: **Sample value:** 4.2	
cxc_api_aggregator_users_logged_out_total Total number of users who are logged out.	**Unit:** **Type:** Gauge **Label:** **Sample value:** 4.2	
cxc_api_aggregator_api_requests_total Total count of requests.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	
cxc_api_healthy_instance Healthy instance.	**Unit:** **Type:** Gauge **Label:** "'ccid', 'tenant_name'" **Sample value:** 4.2	
cxc_api_aggregator_api_requests_processed_success Total count of success requests.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	
cxc_api_aggregator_top_api_requests Top api requests.	**Unit:** **Type:** Counter **Label:** "'path', 'method', 'id', 'name', 'ccid', 'tenant_name', 'code'" **Sample value:** 42	
cxc_api_aggregator_redis_connection_failed Failed Redis connection.	**Unit:** **Type:** Gauge **Label:** "'ccid', 'tenant_name'" **Sample value:** 4.2	
cxc_api_aggregator_request_count Total requests by verb and code.	**Unit:** **Type:** Counter **Label:** "'method', 'path', 'code'" **Sample value:** 42	
cxc_api_aggregator_request_latencies_ms Request latencies histogram by verb, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'path', 'code'" **Sample value:** [1, 2, 3]	
cxc_api_aggregator_request_out_count Total out requests by verb, destination and code.	**Unit:** **Type:** Counter **Label:** "'method', 'destination', 'code'" **Sample value:** 42	
cxc_api_aggregator_request_out_latencies_ms	**Unit:**	

Metric and description	Metric details	Indicator of
Out Request latencies histogram by verb, destination and code, in milliseconds.	**Type:** Histogram **Label:** "'method', 'destination', 'code'" **Sample value:** [1, 2, 3]	
cxc_api_aggregator_elasticsearch_service_latencies_ms Elasticsearch Request latencies histogram by verb, destination and code, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'destination', 'code'" **Sample value:** [1, 2, 3]	

Alerts

The following alerts are defined for API Aggregator.

Alert	Severity	Description	Based on	Threshold
CXC-API-LatencyHigh	HIGH	Triggered when the latency for API responses is beyond the defined threshold.		2500ms for 5m
CXC-API-Redis-Connection-Failed	HIGH	Triggered when the connection to redis fails for more than 1 minute.		1m
CXC-EXT-Ingress-Error-Rate	HIGH	Triggered when the Ingress error rate is above the specified threshold.		20% for 5m
cxc_api_too_many_errors_from_auth	HIGH	Triggered when there are too many error responses from the auth service for more than the specified time threshold.		1m
CXC-CPUUsage	HIGH	Triggered when the CPU utilization of a pod is beyond the threshold		300% for 5m
CXC-MemoryUsage	HIGH	Triggered when the memory utilization of a pod is beyond the threshold.		70% for 5m
CXC-PodNotReadyCount	HIGH	Triggered when the number of pods ready for a CX Contact		1 for 5m

Alert	Severity	Description	Based on	Threshold
		deployment is less than or equal to the threshold.		
CXC-PodRestartsCount	HIGH	Triggered when the restart count for a pod is beyond the threshold.		1 for 5m
CXC-MemoryUsagePD	HIGH	Triggered when the memory usage of a pod is above the critical threshold.		90% for 5m
CXC-PodRestartsCountPD	HIGH	Triggered when the restart count is beyond the critical threshold.		5 for 5m
CXC-PodsNotReadyPD	HIGH	Triggered when there are no pods ready for CX Contact deployment.		0 for 1m

Campaign Manager metrics and alerts

Contents

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

Related documentation:

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
Campaign Manager	ServiceMonitor	3106	/metrics	15 seconds

See details about:

- Campaign Manager metrics
- Campaign Manager alerts

Metrics

Here are some of the metrics exposed by Campaign manager.

Metric and description	Metric details	Indicator of
cxc_cm_campaign_group_created_total Total campaign groups created.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	
cxc_cm_campaign_group_removed_total Total campaign groups removed.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	
cxc_cm_campaign_group_running_total Total campaign groups are running.	**Unit:** **Type:** Gauge **Label:** "'ccid', 'tenant_name'" **Sample value:** 4.2	
cxc_cm_healthy_instance Healthy instance.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_cm_campaign_group_active_total	**Unit:**	

Metric and description	Metric details	Indicator of
Campaign group active total.	**Type:** Gauge **Label:** "'ccid', 'tenant_name'" **Sample value:** 4.2	
cxc_cm_schedule_item_running_total Campaign group with schedule running total.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_cm_schedule_item_active_total Campaign group with schedule active total.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_cm_request_count Total requests by verb and code.	**Unit:** **Type:** Counter **Label:** "'method', 'path', 'code'" **Sample value:** 42	
cxc_cm_request_latencies_ms Request latencies histogram by verb in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'path', 'code'" **Sample value:** [1, 2, 3]	
cxc_cm_request_out_count Total out requests by verb destination and code.	**Unit:** **Type:** Counter **Label:** "'method', 'destination', 'code'" **Sample value:** 42	
cxc_cm_request_out_latencies_ms Out Request latencies histogram by verb destination and code, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'destination', 'code'" **Sample value:** [1, 2, 3]	
cxc_cm_elasticsearch_service_latencies_ms Elasticsearch Request latencies histogram by verb destination and code, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'destination', 'code'" **Sample value:** [1, 2, 3]	

Alerts

The following alerts are defined for Campaign Manager.

Alert	Severity	Description	Based on	Threshold
CXC-CM-Redis-Connection-Failed	HIGH	Triggered when the connection to redis fails for more than		1m

Alert	Severity	Description	Based on	Threshold
		1 minute.		
CXC-CPUUsage	HIGH	Triggered when a the CPU utilization of a pod is beyond the threshold		300% for 5m
CXC-MemoryUsage	HIGH	Triggered when the memory utilization of a pod is beyond the threshold.		70% for 5m
CXC-PodNotReadyCount	HIGH	Triggered when the number of pods ready for a CX Contact deployment is less than or equal to the threshold.		1 for 5m
CXC-PodRestartsCount	HIGH	Triggered when the restart count for a pod is beyond the threshold.		1 for 5m
CXC-MemoryUsagePD	HIGH	Triggered when the memory usage of a pod is above the critical threshold.		90% for 5m
CXC-PodRestartsCountPD	HIGH	Triggered when the restart count is beyond the critical threshold.		5 for 5m
CXC-PodsNotReadyPD	HIGH	Triggered when there are no pods ready for CX Contact deployment.		0 for 1m

Compliance Manager metrics and alerts

Contents

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

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
Compliance Manager	ServiceMonitor	3107	/metrics	15 seconds

See details about:

- Compliance Manager metrics
- Compliance Manager alerts

Metrics

Metric and description	Metric details	Indicator of
compliance_api_history_requests_total Total number of history API calls.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
compliance_validation_under_processing_total Total number validation requests are under processing.	**Unit:** **Type:** Gauge **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 4.2	
compliance_validation_complete_total Total number of completed validation calls.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
compliance_validation_success_total Number of validated requests with Success status.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
compliance_validation_failed_total Number of validation requests with Failed status.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'"	

Metric and description	Metric details	Indicator of
	Sample value: 42	
compliance_validation_success_by_tenant Number of validation requests by Tenant with Success result.	**Unit:** **Type:** Counter **Label:** "'type', 'tenant_name'" **Sample value:** 42	
compliance_validation_failed_by_tenant Number of validation requests by Tenant with Fail result.	**Unit:** **Type:** Counter **Label:** "'type', 'tenant_name'" **Sample value:** 4.2	
cxc_compliance_healthy_instance Healthy instance.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_compliance_request_latencies_ms The latencies of all HTTP requests distributed by method, plus path and HTTP response code.	**Unit:** **Type:** Histogram **Label:** "'method', 'path', 'code'" **Sample value:** [1, 2, 3]	
cxc_compliance_request_count The number of all HTTP requests distributed by method, plus path and HTTP response code.	**Unit:** **Type:** Counter **Label:** "'method', 'path', 'code'" **Sample value:** 42	
compliance_redis_connections_made Total number of Redis connections made.	**Unit:** **Type:** Counter **Label:** n/a **Sample value:** 42	
compliance_redis_connections_closed Total number of Redis connections closed. Current can be calculated with the help of compliance_redis_connections_made.	**Unit:** **Type:** Counter **Label:** n/a **Sample value:** 42	
compliance_redis_access_errors Total number of reported REDIS errors.	**Unit:** **Type:** Counter **Label:** n/a **Sample value:** 42	
compliance_ocs_calls_placed Total number of calls placed by OCS broken by GSW_CALL_RESULT.	**Unit:** **Type:** Counter **Label:** 'GSW_CALL_RESULT' **Sample value:** 42	
cxc_compliance_request_out_count	**Unit:**	

Metric and description	Metric details	Indicator of
Total Out Requests by verb, destination, and code.	**Type:** Counter **Label:** "'method', 'path', 'code'" **Sample value:** 42	
cxc_compliance_request_out_latencies_ms Out Request latencies histogram by verb, destination, and code, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'path', 'code'" **Sample value:** [1, 2, 3]	
cxc_dm_elasticsearch_service_latencies_ms Elasticsearch Request latencies histogram by verb, destination, and code, in milliseconds.	**Unit:** **Type:** Histogram **Label:** n/a **Sample value:** [1, 2. 3]	
cxc_compliance_validation_rate_limit_reached Total number of validation requests rejected due to rate limit exceeded, broken by customer (tenant) and a limit reason {device, customerId, overall}.	**Unit:** **Type:** Counter **Label:** "'ccid', 'reason'" **Sample value:** 42	

Alerts

The following alerts are defined for Compliance Manager.

Alert	Severity	Description	Based on	Threshold
CXC-Compliance-LatencyHigh	HIGH	Triggered when the latency for API responses is beyond the defined threshold.		5000ms for 5m
CXC-CoM-Redis-no-active-connections	HIGH	Triggered when CX Contact compliance has no active redis connection for 2 minutes		2m
CXC-CPUUsage	HIGH	Triggered when the CPU utilization of a pod is beyond the threshold.		300% for 5m
CXC-MemoryUsage	HIGH	Triggered when the memory utilization of a pod is beyond the threshold.		70% for 5m
CXC-PodNotReadyCount	HIGH	Triggered when the number of pods ready for a CX		1 for 5m

Alert	Severity	Description	Based on	Threshold
		Contact deployment is less than or equal to the threshold.		
CXC-PodRestartsCount	HIGH	Triggered when the restart count for a pod is beyond the threshold.		1 for 5m
CXC-MemoryUsagePD	HIGH	Triggered when the memory usage of a pod is above the critical threshold.		90% for 5m
CXC-PodRestartsCountPD	HIGH	Triggered when the restart count is beyond the critical threshold.		5 for 5m
CXC-PodsNotReadyPD	HIGH	Triggered when there are no pods ready for CX Contact deployment.		0 for 1m

Dial Manager metrics and alerts

Contents

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

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
Dial Manager	ServiceMonitor	3109	/metrics	15 seconds

See details about:

- Dial Manager metrics
- Dial Manager alerts

Metrics

Metric and description	Metric details	Indicator of
cxc_dm_healthy_instance Healthy instance.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_dm_processed_batches_total Total processed batches.	**Unit:** **Type:** Counter **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_dm_processed_messages_total Total processed messages.	**Unit:** **Type:** Counter **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_dm_opt_out_messages_total Total opt out messages.	**Unit:** **Type:** Counter **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_dm_failed_processed_messages_total Total failed messages.	**Unit:** **Type:** Counter **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** 42	

Metric and description	Metric details	Indicator of
cxc_dm_batch_size Batch size histogram.	**Unit:** **Type:** Histogram **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** [1, 2, 3]	
cxc_dm_process_message_duration_seconds Processing message duration histogram.	**Unit:** **Type:** Histogram **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** [1, 2, 3]	
cxc_dm_delivery_buffer_size Delivery buffer size.	**Unit:** **Type:** Gauge **Label:** 'media' **Sample value:** 4.2	
cxc_dm_test_messages_total Total test messages.	**Unit:** **Type:** Counter **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_dm_failed_test_messages_total Total failed test messages.	**Unit:** **Type:** Counter **Label:** "'media', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_dm_nexus_service_status The current status of the connection to the Nexus service.	**Unit:** **Type:** Gauge **Label:** "'ccid', 'tenant_name'" **Sample value:** 4.2	
cxc_dm_request_count Total requests made to Nexus via websocket.	**Unit:** **Type:** Counter **Label:** "'media', 'ccid', 'tenant_name', 'code'" **Sample value:** 42	
cxc_dm_request_latencies_ms Request latencies histogram by tenant, in milliseconds.	**Unit:** **Type:** Histogram **Label:** **Sample value:** [1, 2, 3]	
cxc_dm_request_out_count Total out requests by verb, destination, and code.	**Unit:** **Type:** Counter **Label:** "'method', 'destination', 'code'" **Sample value:** 42	
cxc_dm_request_out_latencies_ms Out Request latencies histogram by verb, destination, and code, in milliseconds.	**Unit:** **Type:** Histogram **Label:** **Sample value:** [1, 2, 3]	
cxc_dm_elasticsearch_service_latencies_ms	**Unit:**	

Metric and description	Metric details	Indicator of
Elasticsearch Request latencies histogram by verb, destination, and code, in milliseconds.	**Type:** Histogram **Label:** "'method', 'destination', 'code'" **Sample value:** [1, 2, 3]	

Alerts

The following alerts are defined for Dial Manager.

Alert	Severity	Description	Based on	Threshold
CXC-DM-LatencyHigh	HIGH	Triggered when the latency for dial manager is above the defined threshold.		5000ms for 5m
CXC-CPUUsage	HIGH	Triggered when the CPU utilization of a pod is beyond the threshold		300% for 5m
CXC-MemoryUsage	HIGH	Triggered when the memory utilization of a pod is beyond the threshold.		70% for 5m
CXC-PodNotReadyCount	HIGH	Triggered when the number of pods ready for a CX Contact deployment is less than or equal to the threshold.		1 for 5m
CXC-PodRestartsCount	HIGH	Triggered when the restart count for a pod is beyond the threshold.		1 for 5m
CXC-MemoryUsagePD	HIGH	Triggered when the memory usage of a pod is above the critical threshold.		90% for 5m
CXC-PodRestartsCountPD	HIGH	Triggered when the restart count is beyond the critical threshold.		5 for 5m
CXC-PodsNotReadyPD	HIGH	Triggered when there are no pods ready for CX Contact deployment.		0 for 1m

Job Scheduler metrics and alerts

Contents

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

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
Job Scheduler	ServiceMonitor	3108	/metrics	15 seconds

See details about:

- Job Scheduler metrics
- Job Scheduler alerts

Metrics

Metric and description	Metric details	Indicator of
cxc_js_jobs_executed_total Total jobs executed.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_js_jobs_failed_total Total failed jobs.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_js_jobs_success_total Total successful jobs.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_js_jobs_nothing_to_do_total Total jobs with Nothing TO DO result.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_js_jobs_run_now_total Total jobs that were started manually.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	

Metric and description	Metric details	Indicator of
cxc_js_files_imported_total Total files imported.	**Unit:** **Type:** Counter **Label:** "'action', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_js_jobs_ttl_exceeded_total Total ttl exceeded jobs.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_js_jobs_running_total Number of currently active jobs.	**Unit:** **Type:** Gauge **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** 4.2	
cxc_js_redis_connections Count of active connections to Redis server.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_js_job_duration_seconds Job duration histogram.	**Unit:** **Type:** Histogram **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** [1, 2, 3]	
cxc_js_job_import_file_size_megabytes Job import file size histogram.	**Unit:** **Type:** Histogram **Label:** "'action', 'ccid', 'tenant_name'" **Sample value:** [1, 2, 3]	
cxc_js_healthy_instance Healthy instance.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_js_request_count Total requests by verb and code.	**Unit:** **Type:** Counter **Label:** "'method', 'path', 'code'" **Sample value:** 42	
cxc_js_request_latencies_ms Request latencies histogram by verb, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'path', 'code'" **Sample value:** [1, 2, 3]	
cxc_js_request_out_count Total out requests by verb, destination, and code.	**Unit:** **Type:** Counter **Label:** "'method', 'destination', 'code'" **Sample value:** 42	
cxc_js_request_out_latencies_ms	**Unit:**	

Metric and description	Metric details	Indicator of
Out Request latencies histogram by verb, destination, and code, in milliseconds.	**Type:** Histogram **Label:** "'method', 'destination', 'code'" **Sample value:** [1, 2. 3]	
cxc_js_healthy_tenants Healthy tenants.	**Unit:** **Type:** Gauge **Label:** "'ccid', 'tenant_name'" **Sample value:** 4.2	

Alerts

The following alerts are defined for Job Scheduler.

Alert	Severity	Description	Based on	Threshold
CXC-JS-LatencyHigh	HIGH	Triggered when the latency for job scheduler is above the defined threshold.		5000ms for 5m
CXC-CPUUsage	HIGH	Triggered when the CPU utilization of a pod is beyond the threshold		300% for 5m
CXC-MemoryUsage	HIGH	Triggered when the memory utilization of a pod is beyond the threshold.		70% for 5m
CXC-PodNotReadyCount	HIGH	Triggered when the number of pods ready for a CX Contact deployment is less than or equal to the threshold.		1 for 5m
CXC-PodRestartsCount	HIGH	Triggered when the restart count for a pod is beyond the threshold.		1 for 5m
CXC-MemoryUsagePD	HIGH	Triggered when the memory usage of a pod is above the critical threshold.		90% for 5m
CXC-PodRestartsCountPD	HIGH	Triggered when the restart count is beyond the critical threshold.		5 for 5m
CXC-	HIGH	Triggered when		0 for 1m

Alert	Severity	Description	Based on	Threshold
PodsNotReadyPD		there are no pods ready for CX Contact deployment.		

List Builder metrics and alerts

Contents

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

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
List Builder	ServiceMonitor	3104	/metrics	15 seconds

See details about:

- List Builder metrics
- List Builder alerts

Metrics

Metric and description	Metric details	Indicator of
cxc_lb_jobs_running_total Number of currently active jobs.	**Unit:** **Type:** Counter **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** [1, 2, 3]	
cxc_lb_contacts_imported_total Total contacts imported.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	
cxc_lb_devices_imported_total Total device imported.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	
cxc_lb_rejected_contact_lines_total Total of rejected lines in input contact list files.	**Unit:** **Type:** Counter **Label:** "'ccid', 'tenant_name'" **Sample value:** 42	
cxc_lb_healthy_instance Healthy instance.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	

Metric and description	Metric details	Indicator of
cxc_lb_request_count Total requests by verb and code.	**Unit:** **Type:** Counter **Label:** "'method', 'path', 'code'" **Sample value:** 42	
cxc_lb_request_latencies_ms Request latencies histogram by verb, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'path', 'code'" **Sample value:** [1, 2, 3]	
cxc_lb_job_count Total jobs.	**Unit:** **Type:** Counter **Label:** "'type', 'result', 'ccid', 'tenant_name'" **Sample value:** 42	
cxc_lb_job_duration_seconds Jobs duration histogram in seconds	**Unit:** **Type:** Histogram **Label:** "'type', 'ccid', 'tenant_name'" **Sample value:** [1, 2, 3]	
cxc_lb_request_out_count Total out requests by verb, destination, and code.	**Unit:** **Type:** Counter **Label:** "'method', 'destination', 'code'" **Sample value:** 42	
cxc_lb_request_out_latencies_ms Out Request latencies histogram by verb, destination, and code, in milliseconds.	**Unit:** **Type:** Histogram **Label:** "'method', 'destination', 'code'" **Sample value:** [1, 2, 3]	

Alerts

The following alerts are defined for List Builder.

Alert	Severity	Description	Based on	Threshold
CXC-LB-LatencyHigh	HIGH	Triggered when the latency for list builder is above the defined threshold.		5000ms for 5m
CXC-CPUUsage	HIGH	Triggered when the CPU utilization of a pod is beyond the threshold		300% for 5m
CXC-MemoryUsage	HIGH	Triggered when the memory utilization of a pod is beyond		70% for 5m

Alert	Severity	Description	Based on	Threshold
		the threshold.		
CXC-PodNotReadyCount	HIGH	Triggered when the number of pods ready for a CX Contact deployment is less than or equal to the threshold.		1 for 5m
CXC-PodRestartsCount	HIGH	Triggered when the restart count for a pod is beyond the threshold.		1 for 5m
CXC-MemoryUsagePD	HIGH	Triggered when the memory usage of a pod is above the critical threshold.		90% for 5m
CXC-PodRestartsCountPD	HIGH	Triggered when the restart count is beyond the critical threshold.		5 for 5m
CXC-PodsNotReadyPD	HIGH	Triggered when there are no pods ready for CX Contact deployment.		0 for 1m

List Manager metrics and alerts

Contents

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

Related documentation:

-

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
List Manager	ServiceMonitor	3105	/metrics	15 seconds

See details about:

- List Manager metrics
- List Manager alerts

Metrics

Metric and description	Metric details	Indicator of
cxc_list_manager_executed_jobs_count Total executed jobs count.	**Unit:** **Type:** Counter **Label:** n/a **Sample value:** 42	
cxc_list_manager_running_jobs_count Running jobs count.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_list_manager_rejected_jobs_count Rejected jobs count.	**Unit:** **Type:** Counter **Label:** n/a **Sample value:** 42	
cxc_list_manager_jobs_duration Job duration, in milliseconds.	**Unit:** **Type:** Histogram **Label:** n/a **Sample value:** [1, 2, 3]	
cxc_list_manager_responses_summary Response time, in milliseconds.	**Unit:** **Type:** Summary **Label:** "'method', 'path', 'status'" **Sample value:** 42	

Metric and description	Metric details	Indicator of
cxc_list_manager_healthy_instance Healthy instance.	**Unit:** **Type:** Gauge **Label:** n/a **Sample value:** 4.2	
cxc_list_manager_downloaded_compliance_files_count Count of downloaded compliance files.	**Unit:** **Type:** Counter **Label:** n/a **Sample value:** 42	
cxc_list_manager_contacts_lists_created_count Count of created Contacts Lists.	**Unit:** **Type:** Counter **Label:** "'ccid','tenant_name'" **Sample value:** 42	
cxc_list_manager_import_contacts_requests_processed_count Count of created Contacts Lists.	**Unit:** **Type:** Counter **Label:** "'ccid','tenant_name'" **Sample value:** 42	
cxc_list_manager_import_contacts_requests_failed_count Count of created Contacts Lists.	**Unit:** **Type:** Counter **Label:** "'ccid','tenant_name'" **Sample value:** 42	

Alerts

The following alerts are defined for List Manager.

Alert	Severity	Description	Based on	Threshold
CXC-LM-LatencyHigh	HIGH	Triggered when the latency for list manager is above the defined threshold		5000ms for 5m
cxc_list_manager_too_many_errors_from_auth	HIGH	Triggered when there are too many error responses from the auth service (list manager) for more than the specified time threshold.		1m
CXC-CPUUsage	HIGH	Triggered when the CPU utilization of a pod is beyond the threshold		300% for 5m

Alert	Severity	Description	Based on	Threshold
CXC-MemoryUsage	HIGH	Triggered when the memory utilization of a pod is beyond the threshold.		70% for 5m
CXC-PodNotReadyCount	HIGH	Triggered when the number of pods ready for a CX Contact deployment is less than or equal to the threshold.		1 for 5m
CXC-PodRestartsCount	HIGH	Triggered when the restart count for a pod is beyond the threshold.		1 for 5m
CXC-MemoryUsagePD	HIGH	Triggered when the memory usage of a pod is above the critical threshold.		90% for 5m
CXC-PodRestartsCountPD	HIGH	Triggered when the restart count is beyond the critical threshold.		5 for 5m
CXC-PodsNotReadyPD	HIGH	Triggered when there are no pods ready for CX Contact deployment.		0 for 1m