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.

Interaction Server Private Edition Guide

9/7/2026

Table of Contents

Overview

- About Interaction Server 6
- Architecture 9
- High availability and disaster recovery 15

Configure and deploy

- Before you begin 16
- Configure Interaction Server 20
- Provision Interaction Server 33
- Deploy Interaction Server 34

Upgrade, roll back, or uninstall

- Upgrade, rollback, or uninstall Interaction Server 49

Observability

- Observability in Interaction Server 51
- Interaction Server (IXN) metrics and alerts 54
- Logging 67

Contents

- 1 Overview
- 2 Configure and deploy
- 3 Upgrade, roll back, or uninstall
- 4 Observability

Find links to all the topics in this guide.

Related documentation:

-
-

RSS:

- For private edition

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

Overview

Learn more about Interaction Server, its architecture, and how to support high availability and disaster recovery.

- About Interaction Server
- Architecture
- High availability and disaster recovery

Configure and deploy

Find out how to configure and deploy Interaction Server.

- Before you begin
- Configure Interaction Server
- Provision Interaction Server
- Deploy Interaction Server

Upgrade, roll back, or uninstall

Find out how to upgrade, roll back, or uninstall Interaction Server.

- Upgrade, rollback, or uninstall Interaction Server

Observability

Learn how to monitor Interaction Server with metrics and logging.

- Logging

About Interaction Server

Contents

- 1 IXN Service
  - 1.1 IXN Server
  - 1.2 IXN Node
  - 1.3 IXN VQ Node
- 2 Supported Kubernetes platforms

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

Related documentation:

-
-
-

RSS:

- For private edition

IXN Service

The new Interaction Server (IXN) Service is built basing on the existing IXN Server with few peculiarities only. It consists of the following PODs:

- IXN Node POD
  - IXN Server container
  - IXN Node container (new component)
- IXN VQ Node POD
  - IXN VQ Node container (new component)

IXN Server

Almost all IXN Server functionality is available in IXN Service. Refer to Overview for more information about IXN Server.

IXN Server in container image is configured to work with Postgres DB only. MSSQL and Oracle are not supported now.

IXN Node

IXN Node is provided in IXN Node POD alongside with IXN Server. It's used as a bridge between IXN Server and new Redis-based routing system. It uses IXN Server protocol like old ORS, fetches interactions for routing, sends them for routing via Redis and RQ Service and forwards routing instructions received from routing system to IXN Server.

IXN VQ Node

IXN VQ Node is provided in own IXN VQ Node POD. It's used for sending VQ events from URS to the rest of the system, primarily to Kafka for new Kafka based reporting. StatServers should be

connected to IXN VQ Node to receive VQ events as well.

Supported Kubernetes platforms

IXN service is supported on the following cloud platforms:

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

See the Interaction Server Release Notes for information about when support was introduced.

Architecture

Contents

- 1 Introduction
- 2 Architecture diagram — Connections
- 3 Connections table

Learn about Interaction Server architecture

Related documentation:

-
-
-

RSS:

- For private edition

Introduction

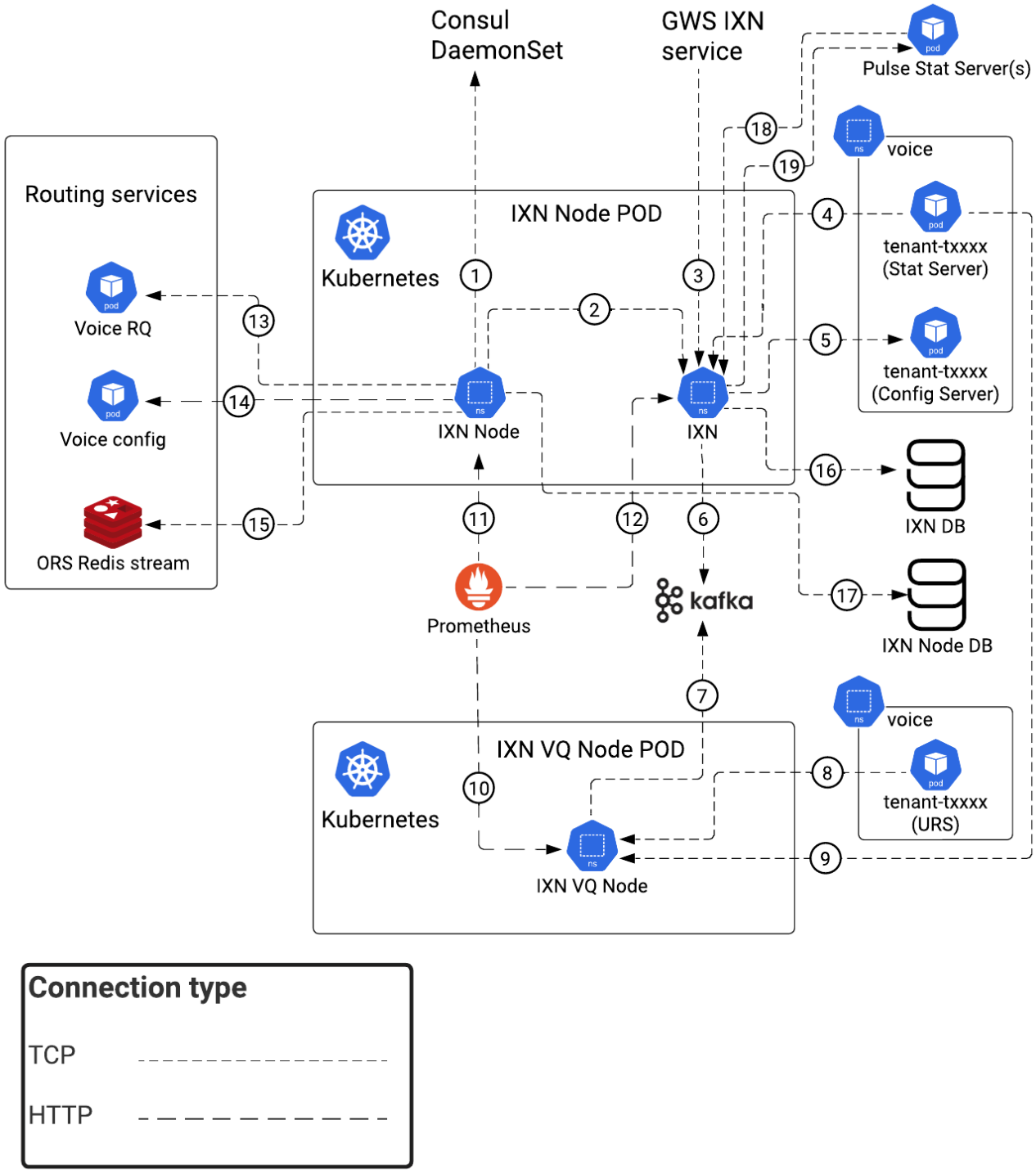
The following diagram displays the architecture for Interaction Server.

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 Interaction Server as a service in the network.


Consul
DaemonSet
GWS IXN
service
Pulse Stat Server(s)
Routing services
Voice RQ
Voice config
ORS Redis stream
Kubernetes
IXN Node POD
IXN Node
IXN
voice
tenant-txxxx
(Stat Server)
tenant-txxxx
(Config Server)
IXN DB
IXN Node DB
Prometheus
kafka
Kubernetes
IXN VQ Node POD
IXN VQ Node
voice
tenant-txxxx
(URS)
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
Connection type
TCP
HTTP

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 Interaction Server as a service in the network. *Egress* means the Interaction Server service is the source, and *Ingress* means the Interaction Server 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	IXN Node	Consul DaemonSet	TCP	8500	Intra-cluster	Configurable parameter: **ixnService.ixnNode.cons**
2	IXN Node	IXN	TCP	7120	Intra-cluster	Configurable parameter: **ixnService.ixnServer.por**
3	GWS IXN service	IXN	TCP	7120	Intra-cluster	Supports all operations via IXN protocol. Configurable parameter: **ixnService.ixnServer.por**
4	StatServer	IXN	TCP	7120	Intra-cluster	Subscribes for reporting information. Configurable parameter: **ixnService.ixnServer.por**
5	IXN	Config Server	TCP	8888	Intra-cluster	Configurable parameter: **ixnService.ixnServer.con**
6	IXN	Kafka	TCP		Egress	IXN Reporting Protocol is streamed to Kafka for consumption by GIM. Port is defined in Kafka configuration.
7	IXN VQ Node	Kafka	TCP		Egress	VQ events are streamed to Kafka for consumption by GIM. Port

Connection	Source	Destination	Protocol	Port	Classification	Data that travels on this connection
						is defined in Kafka configuration.
8	URS	IXN VQ Node	TCP	7122	Intra-cluster	Provides VQ events. Configurable parameter: **ixnVQNode.ports.default**
9	StatServer	IXN VQ Node	TCP	7122	Intra-cluster	Subscribes for VQ events. Configurable parameter: **ixnVQNode.ports.default**
10	Prometheus	IXN VQ Node	HTTP	13139	Ingress	Used for polling of IXN VQ Node metrics endpoint. Configurable parameter: **ixnVQNode.ports.health**
11	Prometheus	IXN Node	HTTP	13133	Ingress	Used for polling of IXN Node metrics endpoint. Configurable parameter: **ixnService.ixnNode.ports**
12	Prometheus	IXN	HTTP	13131	Ingress	Used for polling of IXN metrics endpoint. Configurable parameter: **ixnService.ixnServer.por**
13	IXN Node	Voice RQ	TCP		Intra-cluster	Streams of interaction content messages for ORS. (Port is defined in RQ Service.
14	IXN Node	Voice config	HTTP	8888	Intra-cluster	Access to Config Server data for IXN node

Connection	Source	Destination	Protocol	Port	Classification	Data that travels on this connection
						via Config Node service. Configurable parameter: **ixnService.ixnNode.confi**
15	IXN Node	ORS Redis Stream	TCP		Egress	Stream with interaction IDs to be routed by ORS and stream with routing instructions to IXN. Port is defined in Redis configuration.
16	IXN	IXN DB	TCP		Egress	CRUD operations with Interaction Server database. Port is defined in DBMS configuration.
17	IXN Node	IXN Node DB	TCP		Egress	CRUD operations with IXN Node database. Port is defined in DBMS configuration.
18	Pulse Stat Server(s)	IXN	TCP	7120	Ingress	Subscribes for reporting information. Configurable parameter: **ixnService.ixnServer.por**
19	IXN	Pulse Stat Server(s)	TCP	2060	Egress	Streaming statistics. Port is defined in Config Server.

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

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

Not applicable. Due to its nature, IXN does not support HA/DR. IXN Server service runs single instance per tenant but uses Kubernetes autorestart functionality.

Before you begin

Contents

- 1 Limitations and assumptions
- 2 Download the Helm charts
- 3 Third-party prerequisites
- 4 Storage requirements
- 5 Network requirements
- 6 Browser requirements
- 7 Genesys dependencies

Find out what to do before deploying Interaction Server.

Related documentation:

-
-
-

RSS:

- For private edition

Limitations and assumptions

The current version of IXN Server:

- supports single-region model of deployment only
- does not support scaling or HA
- requires dedicated PostgreSQL deployment per customer

Download the Helm charts

Available IXN containers can be found by the following names in registry:

- *ixn/ixn_vq_node*
- *ixn/ixn_node*
- *ixn/interaction_server*

Available helm charts can be found by the name *ixn-*

For information about downloading Genesys containers and Helm charts from JFrog Edge, see the suite-level documentation: Downloading your Genesys Multicloud CX containers.

Third-party prerequisites

The following are the minimum versions supported by IXN Server:

- Kubernetes 1.17+

- Helm 3.0

Third-party services

Name	Version	Purpose	Notes
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.	
PostgreSQL	11.x	Relational database.	
Kafka	2.x	Message bus.	Kafka is required to deliver IXN reporting events to Genesys Info Mart (GIM). It is the same Kafka for GIM and IXN Server.
Consul	1.8	Service discovery, service mesh, and key/value store.	
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.	Redis is required for communication between Orchestration Server (ORS) and IXN Server. It is the same Redis for ORS and IXN Server.

Storage requirements

In case logging into files is configured for IXN Server, it requires a volume storage mounted to IXN Server container. The storage must be capable to write up to 100 MB/min and 10 MB/s for 2 minutes in peak. The storage size depends on logging configuration.

Regarding storage characteristics for IXN Server database, refer to PostgreSQL documentation.

Contact your account representative if you need assistance with sizing calculations.

Network requirements

Not applicable

Browser requirements

Not applicable

Genesys dependencies

Tenant service. For more information, refer to the Tenant Service Private Edition Guide.

Configure Interaction Server

Contents

- 1 Deployment configuration settings (Helm values)
- 2 Interaction Server deployment settings
- 3 Configuration Server settings
- 4 Security Context

Learn how to prepare and configure deployment of Interaction Server (IXN) using Helm chart.

Related documentation:

-
-
-

RSS:

- For private edition

Deployment configuration settings (Helm values)

Following items should be configured:

- Interaction Server deployment settings
- Configuration Server settings
- Security context

Interaction Server deployment settings

The following table provides information on the Interaction Server deployment settings.

Key	Type	Default	Description
image.imagePullSecrets	list	[]	imagePullSecrets must have the following format: - name: pullSecret1 - name: pullSecret2
image.pullPolicy	string	"IfNotPresent"	Images pull policy
image.registry	string	""	Images registry
ixnService.annotations	string	nil	Extra Annotations ref:Annotations Must be declared with starting |- since they

Key	Type	Default	Description
			are parsed as template
ixnService.image.ixnNode.repository	string	""	Interaction Server Node docker image repository
ixnService.image.ixnNode.tag	string	""	Interaction Server Node docker image repository tag
ixnService.image.ixnServer.repository	string	""	Interaction Server repository
ixnService.image.ixnServer.tag	string	""	Interaction Server tag
ixnService.image.logSidecar.enabled	boolean	false	Enable Interaction Server logging sidecar
ixnService.image.logSidecar.repository	string	""	Interaction Server logging sidecar docker image repository
ixnService.image.logSidecar.tag	string	""	Interaction Server logging sidecar docker image tag
ixnService.ixnNode.configNode.host	string	nil	Interaction Server Node connects to Voice Config Service host
ixnService.ixnNode.configNode.port	int	8888	Interaction Server Node connects to Voice Config Service port
ixnService.ixnNode.consul.host	string	nil	A consul host is either a string literal or valueFrom definition to be inserted into pod definition
ixnService.ixnNode.consul.port	int	8500	A consul port
ixnService.ixnNode.consul.sslMode	string	nil	Connect to Consul using SSL mode: true or false
ixnService.ixnNode.livenessProbe.failureThreshold	int	3	Interaction Server Node

Key	Type	Default	Description
			liveness probe failure threshold
ixnService.ixnNode.livenessProbe.initialDelaySeconds	int	15	Interaction Server Node liveness probe initial delay
ixnService.ixnNode.livenessProbe.periodSeconds	int	30	Interaction Server Node liveness probe check period
ixnService.ixnNode.livenessProbe.timeoutSeconds	int	3	Interaction Server Node liveness probe timeout
ixnService.ixnNode.ports.default	int	13133	Interaction Server Node default port
ixnService.ixnNode.readinessProbe.failureThreshold	int	3	Interaction Server Node readiness probe failure threshold
ixnService.ixnNode.readinessProbe.initialDelaySeconds	int	15	Interaction Server Node readiness probe initial delay
ixnService.ixnNode.readinessProbe.periodSeconds	int	30	Interaction Server Node readiness probe check period
ixnService.ixnNode.readinessProbe.timeoutSeconds	int	3	Interaction Server Node readiness probe timeout
ixnService.ixnNode.redisOptions.tls.enabled	bool	nil	Enable TLS mode for Interaction Server Node to Redis connection
ixnService.ixnNode.redisOptions.tls.rejectUnauthorized	bool	nil	Reject unauthorized hostnames when using TLS connection
ixnService.ixnNode.redis[0]	object	{"host":null,"is_redis_cluster":null,"port":6379}	Interaction Server Node connects to Redis host
ixnService.ixnNode.redis[0].is_redis_cluster	string	nil	Is Redis instance a Cluster or not: true and false

Key	Type	Default	Description
ixnService.ixnNode.redis[0].port	int	6379	Interaction Server Node connects to Redis port
ixnService.ixnNode.resources.limits.cpu	string	"300m"	Interaction Server Node Kubernetes CPU limit
ixnService.ixnNode.resources.limits.memory	string	"320Mi"	Interaction Server Node Kubernetes memory limit
ixnService.ixnNode.resources.requests.cpu	string	"40m"	Interaction Server Node Kubernetes CPU request
ixnService.ixnNode.resources.requests.memory	string	"128Mi"	Interaction Server Node Kubernetes memory request
ixnService.ixnNode.settingsFile	string	"settings.json"	Interaction Server Node settings file name
ixnService.ixnNode.settingsMountPath	string	"/mnt/settings"	Interaction Server Node settings mount path
ixnService.ixnNode.volumeMounts	Object	{}	Volumes mounted into an Interaction Server Node container
ixnService.ixnServer.confServer.appName	string	nil	Interaction Server application name in Configuration Server
ixnService.ixnServer.confServer.host	string	nil	Interaction Server connects to Configuration Server host
ixnService.ixnServer.confServer.port	int	8888	Interaction Server connects to Configuration Server port
ixnService.ixnServer.db.connectionString	string	"KeepaliveInterval=1;KeepaliveTime=60;"	Interaction Server Database connection string suffix
ixnService.ixnServer.db.engine	string	"postgre"	Interaction Server Database engine

Key	Type	Default	Description
ixnService.ixnServer.db.host	string	""	Interaction Server Database host
ixnService.ixnServer.db.name	string	""	Interaction Server Database name
ixnService.ixnServer.db.optionBlobChunkSize	string	nil	Interaction Server Database Blob Chunk Size. Can be left empty
ixnService.ixnServer.db.optionReconnectPause	string	nil	Interaction Server Database Reconnect Pause. Can be left empty
ixnService.ixnServer.db.port	int	5432	Interaction Server Database port
ixnService.ixnServer.db.schemaName	string	nil	Interaction Server Database schema name. Can be left empty
ixnService.ixnServer.jvmOptions.1	string	"-XX:+UnlockExperimentalVMOptions"	
ixnService.ixnServer.jvmOptions.2	string	"-XX:+UseCGroupMemoryLimitForHeap"	
ixnService.ixnServer.jvmOptions.3	string	"-XX:+UseG1GC"	
ixnService.ixnServer.jvmOptions.4	string	"-XX:MinHeapFreeRatio=5"	
ixnService.ixnServer.jvmOptions.5	string	"-XX:MaxHeapFreeRatio=10"	
ixnService.ixnServer.jvmOptions.6	string	"-XX:GCTimeRatio=4"	
ixnService.ixnServer.jvmOptions.7	string	"-XX:AdaptiveSizePol"	
ixnService.ixnServer.livenessProbe.failureThreshold	int	3	Interaction Server liveness probe failure threshold
ixnService.ixnServer.livenessProbe.initialDelaySeconds	int	15	Interaction Server

Key	Type	Default	Description
			liveness probe initial delay
ixnService.ixnServer.livenessProbe.periodSeconds	int	30	Interaction Server liveness probe check period
ixnService.ixnServer.livenessProbe.timeoutSeconds	int	3	Interaction Server liveness probe timeout
ixnService.ixnServer.logStorage.mountPath	string	"/mnt/logs"	Interaction Server logs mount path
ixnService.ixnServer.logStorage.storageClassName	string	nil	Interaction Server log storage class name. Used for PVC, can be left empty
ixnService.ixnServer.logStorage.storageSize	string	nil	Interaction Server log storage size. Used for PVC, can be left empty
ixnService.ixnServer.logStorage.volume	object	{}	A volume definition to be inserted into Interaction Server container definition
ixnService.ixnServer.ports.default	int	13130	Interaction Server default port
ixnService.ixnServer.ports.health	int	13131	Interaction Server health port
ixnService.ixnServer.readinessProbe.failureThreshold	int	3	Interaction Server readiness probe failure threshold
ixnService.ixnServer.readinessProbe.initialDelaySeconds	int	15	Interaction Server readiness probe initial delay
ixnService.ixnServer.readinessProbe.periodSeconds	int	30	Interaction Server readiness probe check period
ixnService.ixnServer.readinessProbe.timeoutSeconds	int	3	Interaction Server

Key	Type	Default	Description
			readiness probe timeout
ixnService.ixnServer.resources.limits.cpu	string	"1.25"	Interaction Server Kubernetes CPU limit
ixnService.ixnServer.resources.limits.memory	string	"6Gi"	Interaction Server Kubernetes memory limit
ixnService.ixnServer.resources.requests.cpu	string	"50m"	Interaction Server Kubernetes CPU request
ixnService.ixnServer.resources.requests.memory	string	"1Gi"	Interaction Server Kubernetes memory request
ixnService.ixnServer.secretsDb.enabled	string	nil	Enable Interaction Server database secret
ixnService.ixnServer.secretsDb.password	string	nil	Interaction Server database password to put in the secret
ixnService.ixnServer.secretsDb.secretName	string	nil	Interaction Server database secret name
ixnService.ixnServer.secretsDb.username	string	nil	Interaction Server database username to put in the secret
ixnService.ixnServer.serviceAccount.create	bool	true	Create service account for Interaction Server
ixnService.ixnServer.serviceAccount.name	string	nil	The name of the ServiceAccount to use. If not set and create is true, a name is generated using the fullname template
ixnService.ixnServer.startupProbe.failureThreshold	int	120	Interaction Server startup probe failure threshold
ixnService.ixnServer.startupProbe.periodSeconds	int	30	Interaction Server startup probe check

Key	Type	Default	Description
			period
ixnService.ixnServer.volumeMounts	object	{}	Volumes mounted into an Interaction Server container
ixnService.labels	object	{}	Extra labels ref:Labels and Selectors
ixnService.logSidecar.portsinit	health	13137	Log Sidecar health port
ixnService.logSidecar.resources	string	limits.cpu "300m"	Interaction Server Log Sidecar Kubernetes CPU limit
ixnService.logSidecar.resources	string	limits.memory "320Mi"	Interaction Server Log Sidecar Kubernetes memory limit
ixnService.logSidecar.resources	string	requests.cpu "40m"	Interaction Server Log Sidecar Kubernetes CPU request
ixnService.logSidecar.resources	string	requests.memory "128Mi"	Interaction Server Log Sidecar Kubernetes memory request
ixnService.logSidecar.settings	log	enableHealthPort false	Log Sidecar enable health port
ixnService.nodeSelector	object	{}	Node labels for assignment. ref:Labels and Selectors
ixnService.priorityClassName	string	""	Priority Class ref:Assigning Pods to Nodes
ixnService.prometheus.podMonitor	enabled	false	Enable Promethes PodMonitor
ixnService.prometheus.podMonitor	targetLabels	{}	Promethes PodMonitor target labels
ixnService.securityContext	object	{}	Security Context

Key	Type	Default	Description
			ref:Set the security context for a Container Containers should run as genesys user and cannot use elevated permissions !!! THESE OPTIONS SHOULD NOT BE CHANGED UNLESS INSTRUCTED BY GENESYS !!!
ixnService.service.enabled	bool	true	Enable Kubernetes service for Interaction Service
ixnService.volumes	string	nil	Volumes provided to Interaction Service pod Must be declared with starting |- since they are parsed as a template
ixnVQNode.annotations	string	nil	Extra Annotations ref:Annotations Must be declared with starting |- since they are parsed as a template
ixnVQNode.image.ixnVQNode.repository	string	nil	Interaction Server VQ Node docker image repository
ixnVQNode.image.ixnVQNode.tag	string	nil	Interaction Server VQ Node docker image tag
ixnVQNode.labels	object	{}	Interaction Server VQ Node extra labels ref:Labels and Selectors
ixnVQNode.livenessProbe.failureThreshold	int	3	Interaction Server VQ Node liveness probe failure threshold
ixnVQNode.livenessProbe.initialDelaySeconds	int	15	Interaction Server VQ Node liveness probe initial delay

Key	Type	Default	Description
ixnVQNode.livenessProbe.periodSeconds	int	30	Interaction Server VQ Node liveness probe check period
ixnVQNode.livenessProbe.timeoutSeconds	int	3	Interaction Server VQ Node liveness probe timeout
ixnVQNode.nodeSelector	object	{}	Node labels for assignment. ref:Assigning Pods to Nodes
ixnVQNode.ports.default	int	13138	Interaction Server VQ Node default port
ixnVQNode.ports.health	int	13139	Interaction Server VQ Node health port
ixnVQNode.priorityClassName	string	""	Priority Class ref:Pod Priority and Preemption
ixnVQNode.prometheus.podMonitor.enabled	bool	false	Enable Promethes PodMonitor
ixnVQNode.prometheus.podMonitor.targetLabels	object	{}	Promethes PodMonitor target labels
ixnVQNode.readinessProbe.failureThreshold	int	3	Interaction Server VQ Node readiness probe failure threshold
ixnVQNode.readinessProbe.initialDelaySeconds	int	15	Interaction Server VQ Node readiness probe initial delay
ixnVQNode.readinessProbe.periodSeconds	int	30	Interaction Server VQ Node readiness probe check period
ixnVQNode.readinessProbe.timeoutSeconds	int	3	Interaction Server VQ Node readiness probe timeout
ixnVQNode.resources.limits.cpu	string	"300m"	Interaction Server VQ Node Kubernetes CPU limit

Key	Type	Default	Description
ixnVQNode.resources.limits.memory	string	"160Mi"	Interaction Server VQ Node Kubernetes memory limit
ixnVQNode.resources.requests.cpu	string	"30m"	Interaction Server VQ Node Kubernetes CPU request
ixnVQNode.resources.requests.memory	string	"128Mi"	Interaction Server VQ Node Kubernetes memory request
ixnVQNode.securityContext	object	{}	Security Context ref: Set the security context for a Container Containers should run as genesys user and cannot use elevated permissions !!! THESE OPTIONS SHOULD NOT BE CHANGED UNLESS INSTRUCTED BY GENESYS !!!
ixnVQNode.serviceAccount.create	bool	true	Create service account for Interaction Server VQ Node
ixnVQNode.serviceAccount.name	string	nil	The name of the ServiceAccount to use. If not set and create is true, a name is generated using the fullname template
ixnVQNode.volumeMounts	object	{}	Volumes mounted into an Interaction Server VQ Node container
podAnnotations	object	{}	Add annotations to all pods
podLabels	object	{}	Add labels to all pods
replicaCount	int	1	Replica count. Applied to both Interaction Service and Interaction Server VQ node pods.

Key	Type	Default	Description
			Can only be 1 or 0. Other values are not supported
tenant.id	string	nil	Tenant UUID or GWS ID
tenant.sid	string	nil	Tenant short 4-digit ID

For information on Interaction Server options, refer to the Configuration Options Reference Manual.

Configuration Server settings

Ensure that IXN Server application in Configuration Server has correct health endpoint:

```
Applications -  - Options - [health-service] soap-endpoint = http://localhost:9100/health
```

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
```

Provision Interaction Server

- Administrator

Learn how to provision Interaction Server.

Related documentation:

-
-
-

RSS:

- For private edition

By default, IXN Service creates IXN Server DB if the DB does not exist yet. It's done automatically on pod initialization phase. No manual actions are needed unless this is undesirable for some reason.

All data is stored in PostgreSQL, which is external to the IXN.

IXN requires two PostgreSQL databases to be created: for IXN and IXN Node.

```
ixnService:
  ixnServer:
    secrets:
      db:
        enabled: true
        secretName:
        username:
        password:
  ixnNode:
    secrets:
      db:
        enabled: true
        secretName:
        username:
        password:
```

Deploy Interaction Server

Contents

- 1 Assumptions
- 2 Deploy
  - 2.1 Environment setup
- 3 Prepare cluster resources
  - 3.1 Create secrets
  - 3.2 Service account
- 4 Deploy IXN via Helm
  - 4.1 **IXN Server**
  - 4.2 **ixnNode**
  - 4.3 **Tenant:**
  - 4.4 Log storage
  - 4.5 Consul connection
  - 4.6 Volume mounts
- 5 Configure monitoring and logging
- 6 Validate the deployment

Learn how to deploy Interaction Server (IXN) 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.

Deploy

> **Important**
> Make sure to review Before you begin for the full list of prerequisites required to deploy Interaction Server.

Environment setup

GKE

1. Log in to the gke cluster.

   ```
   gcloud container clusters get-credentials gke1
   ```

2. Create ixn project in the GKE cluster using the following manifest file:

```
Create Interaction Server project in gke cluster using following manifest file:
'''create-ixn-namespace.json'''
{
"apiVersion": "v1",
"kind": "Namespace",
"metadata": {
"name": "ixn",
"labels": {
"name": "ixn"
}
}
}
kubectl apply -f apply create-ixn-namespace.json
```

3. Confirm the namespace creation.

```
kubectl describe namespace ixn
```

4. (Optional step) Create a secret for docker-registry in order to pull image from JFrog.

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

AKS

1. Log in to the AKS cluster

```
az aks get-credentials --resource-group $RESOURCE_GROUP --name $AKS_CLUSTER_NAME
```

2. Create ixn project in the AKS cluster using the following manifest file:

```
{
"apiVersion": "v1",
"kind": "Namespace",
"metadata": {
"name": "ixn",
"labels": {
"name": "ixn"
}
}
}
kubectl apply -f apply create-ixn-namespace.json
```

3. Confirm the namespace creation.

```
kubectl describe namespace ixn
```

Prepare cluster resources

To prepare your resources, create secrets and a default pull secret for the cluster.

Create secrets

Create Kubernetes (K8s) secrets for Redis and Kafka access in the IXN namespace:

```
 kubectl delete secret redis-ors-secret --ignore-not-found

 kubectl create secret generic redis-ors-secret \
 --from-literal='voice-redis-ors-
stream={"password":"PaSSword","port":"1234","rejectUnauthorized":"false","servername":"redis-
cluster.namespace.svc.cluster.local"}'

 kubectl delete secret kafka-shared-secret --ignore-not-found

 kubectl create secret generic kafka-shared-secret \
 --from-literal='kafka-secrets={"bootstrap": "infra-kafka-cp-
kafka.infra.svc.cluster.local:9092"}'
```

The following is a case when username and password are needed for Kafka authentication. A Kubernetes secret creation command will look like this:

```
kubectl create secret generic kafka-shared-secret \

--from-literal='kafka-secrets={"bootstrap": "kafka-service.kafka.svc.cluster.local:9092",
"username":"…", "password":"…"}'
```

Service account

Either create a service account and set it in Helm values file or just modify an existing one after Helm is installed and service account is created.

Here is an example of created service account, it must be named as ixn-server- for consul injection working.

```
kubectl get serviceaccounts
NAME                                          SECRETS   AGE
ixn-server-
```

Deploy IXN via Helm

To deploy IXN via Helm, follow these steps:

1. Download the latest version of Interaction Server installation Helm Charts from the artifactory. See the JFrog Platform Artifactory.

2. Extract parameters from the chart to see multiple (default) values used to fine-tune the installation.

```
$ helm show values /ixn > override_values.yaml
```

Configure the following key entries in the IXN override_values.yaml file:

IXN Server

Secrets:

```
ixnServer:
```

```
  secrets:
    db:
      # -- Enable Interaction Server database secret
      enabled: true
      # -- Interaction Server database secret name
      secretName: ixn-db-secret
      # -- Interaction Server database username to put in the secret
      username: ""
      # -- Interaction Server database password to put in the secret
      password: ""
```

Database:

```
db:
    # -- Interaction Server Database engine
    engine: "postgre"
    # -- Interaction Server Database name
    name: "ixn-db"
    # -- Interaction Server Database host
    host:
    # -- Interaction Server Database port
    port: 5432
    # -- Interaction Server Database connection string suffix
```

ixnNode

Database:

```
  db:
    # -- Interaction Server Node DB host
    host:
    # -- Interaction Server Node DB port
    port: 5432
    # -- Interaction Server Node DB name
    name: ixn-node
```

Redis:

```
  redis:
    # -- Interaction Server Node connects to Redis host
  - host:
    port:
    # -- Is Redis instance a Cluster or not
    is_redis_cluster: "true"
```

Secrets:

```
secrets:
    db:
      # -- Enable Interaction Server Node database secret
      enabled: true
      # -- Interaction Server Node database secret name
      secretName: ixn-node-db-secret
      # -- Interaction Server Node database username to put in the secret
      username: ""
      # -- Interaction Server Node database password to put in the secret
      password: ""
```

Tenant:

```
tenant:
  # -- Tenant UUID or GWS ID
  id: ""
  # -- Tenant short ID
  sid:
```

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

The following is a sample override_values.yaml file. (Also, refer to Log storage, Consul connection, and Volume mounts.)

```
# -- Add labels to all pods
podLabels: {}

# -- Add annotations to all pods
podAnnotations: {}

image:
  # -- Images registry
  registry: "pureengage-docker-staging.jfrog.io"
  # -- Images pull policy
  pullPolicy: Always #IfNotPresent
  # -- imagePullSecrets must have the following format:

  # - name: pullSecret1

  # - name: pullSecret2
  imagePullSecrets:
    - name: pullsecret
    #- name: jfrog-stage-credentials

ixnService:
  image:
    ixnServer:  #see versions.yaml
      # -- Interaction Server repository
      #repository: "ixn/interaction_server"
      # -- Interaction Server tag
      #tag: "latest"

    logSidecar:
      # -- Enable Interaction Server logging sidecar
      enabled: true
      # -- Interaction Server logging sidecar docker image repository
      repository: "fluent/fluent-bit"
      # -- Interaction Server logging sidecar docker image tag
      tag: "1.8.5"

    ixnNode:  #see versions.yaml
      # -- Interaction Server Node docker image repository
      #repository: "ixn/ixn_node"
      # -- Interaction Server Node docker image repository tag
      #tag: "latest"

  service:
    # -- Enable Kubernetes service for Interaction Service
    enabled: true

  # -- Volumes provided to Interaction Service pod
  # Must be declared with starting `\|-`
```

```
  # since they are parsed as a template
  volumes: |-
    - name: redis-ors-secret
      secret:
        secretName: redis-ors-secret
    - name: kafka-shared-secret
      secret:
        secretName: kafka-shared-secret

  # -- Security Context
  # ref: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/#set-the-
security-context-for-a-container
  # Containers should run as genesys user and cannot use elevated permissions
  # !!! THESE OPTIONS SHOULD NOT BE CHANGED UNLESS INSTRUCTED BY GENESYS !!!
  securityContext: {}

  # -- Priority Class
  # ref: https://kubernetes.io/docs/concepts/configuration/pod-priority-preemption/
  priorityClassName: ""

  # -- Node labels for assignment.
  # ref: https://kubernetes.io/docs/user-guide/node-selection/
  nodeSelector: {}

  # -- Extra labels
  # ref: https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/
  labels: {}

  # -- Extra Annotations
  # ref: https://kubernetes.io/docs/concepts/overview/working-with-objects/annotations/
  # Must be declared with starting `\|-`
  # since they are parsed as template
  annotations: |-
      "consul.hashicorp.com/connect-inject": "true"
      "consul.hashicorp.com/connect-service": {{ include "ixn.consulIxnServerName" . | quote
}}
      "consul.hashicorp.com/connect-service-port": server-default
      "consul.hashicorp.com/connect-service-upstreams": |-
        voice-config:{{ .Values.ixnService.ixnNode.configNode.port }},
        {{ printf "tenant-%s:%d" .Values.tenant.id (int
.Values.ixnService.ixnServer.confServer.port) }}
      consul.hashicorp.com/service-tags: 'service-ixn'
      consul.hashicorp.com/service-meta-tenant-id: {{ .Values.tenant.id }}
      consul.hashicorp.com/service-meta-tenant-sid: {{ .Values.tenant.sid | quote }}

  prometheus:
    monitoringService:
      # -- Enable a service with Prometheus annotations for metrics scraping
      enabled: true

  ixnServer:

    serviceAccount:
      # -- Create service account for Interaction Server
      create: true
      # -- The name of the ServiceAccount to use.
      # If not set and create is true, a name is generated using the fullname template
      name:

    ports:
      # -- Interaction Server default port
      default: 7120
      # -- Interaction Server health port
```

```
    health: 9100

  secrets:
    db:
      # -- Enable Interaction Server database secret
      enabled: true
      # -- Interaction Server database secret name
      secretName: ixn-db-secret-${TENANT_ID}
      # -- Interaction Server database username to put in the secret
      username: "${POSTGRES_USER}"
      # -- Interaction Server database password to put in the secret
      password: "${POSTGRES_PASSWORD}"

  confServer:
    # -- Interaction Server connects to Configuration Server host
    host: "localhost"
    # -- Interaction Server connects to Configuration Server port
    port: 8888
    # -- Interaction Server application name in Configuration Server
    appName: InteractionServer

  db:
    # -- Interaction Server Database engine
    engine: "postgre"
    # -- Interaction Server Database name
    name: "ixn-${TENANT_ID}"
    # -- Interaction Server Database host
    host: ${POSTGRES_ADDR}
    # -- Interaction Server Database port
    port: 5432
    # -- Interaction Server Database connection string suffix
    connectionString: "KeepaliveInterval=1;KeepaliveTime=60;"
    # -- Interaction Server Database Blob Chunk Size. Can be left empty
    optionBlobChunkSize:
    # -- Interaction Server Database Reconnect Pause. Can be left empty
    optionReconnectPause:
    # -- Interaction Server Database schema name. Can be left empty
    schemaName:

  dbinit:
    enabled: true

  logStorage:
    # -- Interaction Server logs mount path
    mountPath: "/mnt/logs"
    # -- Interaction Server log storage size. Used for PVC, can be left empty
    storageSize: 1Gi
    # -- Interaction Server log storage class name. Used for PVC, can be left empty
    storageClassName:
    # -- A volume definition to be inserted into Interaction Server container definition
    volume:
      emptyDir: {}

  livenessProbe:
    # -- Interaction Server liveness probe initial delay
    initialDelaySeconds: 15
    # -- Interaction Server liveness probe check period
    periodSeconds: 30
    # -- Interaction Server liveness probe timeout
    timeoutSeconds: 3
    # -- Interaction Server liveness probe failure threshold
    failureThreshold: 3
```

```
    readinessProbe:
      # -- Interaction Server readiness probe initial delay
      initialDelaySeconds: 15
      # -- Interaction Server readiness probe check period
      periodSeconds: 30
      # -- Interaction Server readiness probe timeout
      timeoutSeconds: 3
      # -- Interaction Server readiness probe failure threshold
      failureThreshold: 3

    startupProbe:
      # -- Interaction Server startup probe check period
      periodSeconds: 30
      # -- Interaction Server startup probe failure threshold
      failureThreshold: 120

    resources:
      requests:
        # -- Interaction Server Kubernetes CPU request
        cpu: "100m"
        # -- Interaction Server Kubernetes memory request
        memory: "512Mi"
      limits:
        # -- Interaction Server Kubernetes CPU limit
        cpu: "200m"
        # -- Interaction Server Kubernetes memory limit
        memory: "2Gi"

    jvmOptions:
      1: "-XX:+UnlockExperimentalVMOptions"
      2: "-XX:+UseCGroupMemoryLimitForHeap"
      3: "-XX:+UseG1GC"
      4: "-XX:MinHeapFreeRatio=5"
      5: "-XX:MaxHeapFreeRatio=10"
      6: "-XX:GCTimeRatio=4"
      7: "-XX:AdaptiveSizePol"

    # -- Volumes mounted into an Interaction Server container
    volumeMounts:
      kafka-shared-secret:
        readOnly: true
        mountPath: "/mnt/env-secrets/kafka-secrets"

  ixnNode:

    settings:
      # -- Interaction Server Node settings mount path
      mountPath: "/mnt/settings"
      # -- Interaction Server Node settings file name
      file: "settings.json"

    storingSessions:
      # -- Enable storing Interaction Server Node sessions in database
      enabled: true
    db:
      # -- Interaction Server Node DB host
      host: ${POSTGRES_ADDR}
      # -- Interaction Server Node DB port
      port: 5432
      # -- Interaction Server Node DB name
      name: ixn-node-${TENANT_ID}
      options:
        # -- Keep Interaction Server Node db connection alive
```

```
        keepAlive: true
        # -- Keep Interaction Server Node db connection alive: initial delay in milliseconds
        keepAliveInitialDelayMillis: 300000
        # -- Interaction Server Node db connection ssl options.
        # Details: https://nodejs.org/docs/latest-v14.x/api/
tls.html#tls_new_tls_tlssocket_socket_options
        # If no any specific TLS options are required, yet connection should be established
via TLS,
        # just "ssl" property with empty object as value is required, like ssl: {}
        # here we made empty ssl: to disable it since ssl disabled for OC Postgres
        ssl:

    ports:
      # -- Interaction Server Node default port
      default: 6120

    configNode:
      # -- Interaction Server Node connects to config server host
     host: "localhost"
      # -- Interaction Server Node connects to config server port
      port: 11100

    redis:
      # -- Interaction Server Node connects to Redis host
    - host: ${REDIS_ADDR}
      port: ${REDIS_PORT}
      # -- Is Redis instance a Cluster or not
      is_redis_cluster: "true"

    secrets:
      db:
        # -- Enable Interaction Server Node database secret
        enabled: true
        # -- Interaction Server Node database secret name
        secretName: ixn-node-db-secret-${TENANT_ID}
        # -- Interaction Server Node database username to put in the secret
        username: ${POSTGRES_USER}
        # -- Interaction Server Node database password to put in the secret
        password: "${POSTGRES_PASSWORD}"

    dbinit:
      enabled: true

    redisOptions:
      tls:
        # -- Enable TLS mode for Interaction Server Node to Redis connection
        enabled: false
        # -- Reject unauthorized hostnames when using TLS connection
        rejectUnauthorized: false

    consul:
      # -- A consul host is either a string literal or valueFrom definition to be inserted
into pod definition
      host:
        valueFrom:
          fieldRef:
            fieldPath: status.hostIP
      # -- Consul HTTP port
      port: 8500
      # -- Connect to Consul using SSL mode: true or false
      sslMode: false

    livenessProbe:
```

```
      # -- Interaction Server Node liveness probe initial delay
      initialDelaySeconds: 15
      # -- Interaction Server Node liveness probe check period
      periodSeconds: 30
      # -- Interaction Server Node liveness probe timeout
      timeoutSeconds: 3
      # -- Interaction Server Node liveness probe failure threshold
      failureThreshold: 3

    readinessProbe:
      # -- Interaction Server Node readiness probe initial delay
      initialDelaySeconds: 15
      # -- Interaction Server Node readiness probe check period
      periodSeconds: 30
      # -- Interaction Server Node readiness probe timeout
      timeoutSeconds: 3
      # -- Interaction Server Node readiness probe failure threshold
      failureThreshold: 3

    resources:
      requests:
        # -- Interaction Server Node Kubernetes CPU request
        cpu: "40m"
        # -- Interaction Server Node Kubernetes memory request
        memory: "128Mi"
      limits:
        # -- Interaction Server Node Kubernetes CPU limit
        cpu: "300m"
        # -- Interaction Server Node Kubernetes memory limit
        memory: "320Mi"

    # -- Volumes mounted into an Interaction Server Node container
    volumeMounts:
      redis-ors-secret:
        readOnly: true
        mountPath: "/mnt/env-secrets/redis-secrets"

    # env:
    #   DEBUG: ioredis:*

ixnVQNode:
  image:
    ixnVQNode:  #see versions.yaml
      # -- Interaction Server VQ Node docker image repository
      #repository: "ixn/ixn_vq_node"
      # -- Interaction Server VQ Node docker image tag
      #tag: "latest"

  serviceAccount:
    # -- Create service account for Interaction Server VQ Node
    create: true
    # -- The name of the ServiceAccount to use.
    # If not set and create is true, a name is generated using the fullname template
    name:

  ports:
    # -- Interaction Server VQ Node default port
    default: 7122
    # -- Interaction Server VQ Node health port
    health: 9102

  resources:
    requests:
```

```
      # -- Interaction Server VQ Node Kubernetes CPU request
      cpu: "30m"
      # -- Interaction Server VQ Node Kubernetes memory request
      memory: "128Mi"
    limits:
      # -- Interaction Server VQ Node Kubernetes CPU limit
      cpu: "300m"
      # -- Interaction Server VQ Node Kubernetes memory limit
      memory: "160Mi"

  # -- Security Context
  # ref: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/#set-the-
security-context-for-a-container
  # Containers should run as genesys user and cannot use elevated permissions
  # !!! THESE OPTIONS SHOULD NOT BE CHANGED UNLESS INSTRUCTED BY GENESYS !!!
  securityContext: {}

  # -- Priority Class
  # ref: https://kubernetes.io/docs/concepts/configuration/pod-priority-preemption/
  priorityClassName: ""

  # -- Node labels for assignment.
  # ref: https://kubernetes.io/docs/user-guide/node-selection/
  nodeSelector: {}

  # -- Interaction Server VQ Node extra labels
  # ref: https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/
  labels: {}

  # -- Extra Annotations
  # ref: https://kubernetes.io/docs/concepts/overview/working-with-objects/annotations/
  # Must be declared with starting `\|-`
  # since they are parsed as a template
  annotations: |-
    "consul.hashicorp.com/connect-inject": "true"
    "consul.hashicorp.com/connect-service": {{ include "ixn.consulIxnVQNodeName" . | quote }}
    "consul.hashicorp.com/connect-service-port": vqnode-default

  prometheus:
    monitoringService:
      # -- Enable a service with Prometheus annotations for metrics scraping
      enabled: true

  livenessProbe:
    # -- Interaction Server VQ Node liveness probe initial delay
    initialDelaySeconds: 15
    # -- Interaction Server VQ Node liveness probe check period
    periodSeconds: 30
    # -- Interaction Server VQ Node liveness probe timeout
    timeoutSeconds: 3
    # -- Interaction Server VQ Node liveness probe failure threshold
    failureThreshold: 3

  readinessProbe:
    # -- Interaction Server VQ Node readiness probe initial delay
    initialDelaySeconds: 15
    # -- Interaction Server VQ Node readiness probe check period
    periodSeconds: 30
    # -- Interaction Server VQ Node readiness probe timeout
    timeoutSeconds: 3
    # -- Interaction Server VQ Node readiness probe failure threshold
    failureThreshold: 3
```

```
  # -- Volumes provided to Interaction Server VQ Node pod
  # Must be declared with starting `\|-`
  # since they are parsed as a template
  volumes : |-
    - name: kafka-shared-secret
      secret:
        secretName: kafka-shared-secret

  # -- Volumes mounted into an Interaction Server VQ Node container
  volumeMounts:
    kafka-shared-secret:
      readOnly: true
      mountPath: "/mnt/env-secrets/kafka-secrets"

tenant:
  # -- Tenant UUID or GWS ID
  id: "${TENANT_UUID}"
  # -- Tenant short ID
  sid: ${TENANT_ID}

# -- Replica count. Applied to both Interaction Service and Interaction Server VQ node pods.
# Can only be 1 or 0. Other values are not supported
replicaCount: 1
```

Replace some parameters-placeholders in this file with proper values. Adjust and copy/paste shell variables below (example, might be different in your environment):

```
export TENANT_ID=100
export TENANT_UUID=9350e2fc-a1dd-4c65-8d40-1f75a2e080dd
export POSTGRES_USER=postgres
export POSTGRES_PASSWORD=password
export REDIS_ADDR=infra-redis-redis-cluster.infra
export REDIS_PORT=6379
```

Now, substitute placeholders with these values in override_values.yaml:

```
envsubst  override_values.yaml_
```

Note: It creates a separate file “override_values.yaml_” that you will use in deployment. 3. Validate the Helm chart and provided values:

```
$ helm template ixn-{short-tenant-id} /ixn --version={version} -f override_values.yaml_
```

4. Install the Interaction Server chart, using the override values file:

```
$ helm upgrade --install ixn-{short-tenant-id} /ixn --version={version} -f
override_values.yaml_
```

5. Wait until all containers are ready. There should be 4/4 (5/5 if a logging sidecar enabled) for ixn-*-sts-0 and 3/3 containers for ixn-*-vqnode. If it is 1/1, it usually means something is wrong with the consul sidecar injection.

If the following error appeared: "*line 5: exec: /home/genesys/interaction_server/interaction_server_64: cannot execute: Permission denied*", ixn-{short-tenant-id}-sts-0 pod restart may be required if service account policy was applied after pod started). Refer to Service account.

```
kubectl get pods
NAME                                              READY   STATUS    RESTARTS   AGE
ixn-{short-tenant-id}-sts-0                       4/4     Running  0          8m1s
ixn-{short-tenant-id}-vqnode-deploy-6d8bc6846d-ml49d  3/3     Running  0          21m
```

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

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

```
$ helm show values /ixn > override_values.yaml_
```

Log storage

The following is a log storage example configuration in IXN Helm values:

```
ixnService:
  ixnServer:
    logStorage:
      mountPath: "/mnt/logs"
      storageSize: 1Gi
      storageClassName:
      volume:
        emptyDir: {}
```

Consul connection

Consul connection can be configured in several ways:

```
ixnService:
  ixnNode:
    consul:
      host:
        value:
      port:
      sslMode: false
```

```
ixnService:
  ixnNode:
    consul:
      host:
        valueFrom:
          fieldRef:
            fieldPath: status.hostIP
      port:
      sslMode: false
```

Connection to Configuration Server using Consul

```
ixnService:
  annotations: |-
      "consul.hashicorp.com/connect-inject": "true"
      "consul.hashicorp.com/connect-service": {{ include "ixn.consulIxnServerName" . | quote
}}
      "consul.hashicorp.com/connect-service-port": server-default
      "consul.hashicorp.com/connect-service-upstreams": |-
        voice-config:{{ .Values.ixnService.ixnNode.configNode.port }},
        {{ printf "tenant-%s:%d" .Values.tenant.id (int
.Values.ixnService.ixnServer.confServer.port) }}
      consul.hashicorp.com/service-tags: 'service-ixn'
      consul.hashicorp.com/service-meta-tenant-id: {{ .Values.tenant.id }}
      consul.hashicorp.com/service-meta-tenant-sid: {{ .Values.tenant.sid | quote }}
  ixnServer:
    confServer:
      host: "localhost"
```

```
      port: 8888
      appName: InteractionServer
  ixnNode:
    configNode:
      host: "localhost"
      port: 11100
ixnVQNode:
  annotations: |-
    "consul.hashicorp.com/connect-inject": "true"
    "consul.hashicorp.com/connect-service": {{ include "ixn.consulIxnVQNodeName" . | quote }}
    "consul.hashicorp.com/connect-service-port": vqnode-default
```

Volume mounts

Volume mounts example:

```
ixnService:
  volumes: |-
    - name: redis-ors-secret
      secret:
        secretName: redis-ors-secret
    - name: kafka-shared-secret
      secret:
        secretName: kafka-shared-secret
  ixnServer:
    volumeMounts:
      kafka-shared-secret:
        readOnly: true
        mountPath: "/mnt/env-secrets/kafka-secrets"
  ixnNode:
    volumeMounts:
      redis-ors-secret:
        readOnly: true
        mountPath: "/mnt/env-secrets/redis-secrets"
  ixnVQNode:
    volumes : |-
      - name: kafka-shared-secret
        secret:
          secretName: kafka-shared-secret
    volumeMounts:
      kafka-shared-secret:
        readOnly: true
        mountPath: "/mnt/env-secrets/kafka-secrets"
```

Configure monitoring and logging

To configure monitoring parameters in the Helm values file, see Monitoring.

To configure logging parameters in the Helm values file, see Logging.

Validate the deployment

There must be two pods. Each pod must be in a Running state and all READY checks should pass.

Upgrade, rollback, or uninstall Interaction Server

Contents

- 1 Upgrade Interaction Server
- 2 Rollback Interaction Server
- 3 Uninstall Interaction Server

Learn how to upgrade, rollback or uninstall Interaction Server.

Related documentation:

-
-
-

RSS:

- For private edition

Upgrade Interaction Server

Update Docker image versions in the values file or get a new Helm chart. Then install it with the following command:

```
helm upgrade --install   -f
```

Rollback Interaction Server

Set previous Docker image versions in the values file or get a previous Helm chart. Then install it with the following command:

```
helm upgrade --install   -f
```

Uninstall Interaction Server

```
helm uninstall
```

Manually delete all external secrets, if needed.

Observability in Interaction Server

Contents

- 1 Monitoring
  - 1.1 Enable monitoring
  - 1.2 Configure metrics
- 2 Alerting
- 3 Logging

Learn about the logs, metrics, and alerts you should monitor for Interaction Server.

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 Interaction Server metrics, see:

- Interaction Server (IXN) metrics

See also System metrics.

Enable monitoring

Monitoring is not enabled by default. To enable monitoring, modify the Helm chart by setting the following values in the values.yaml file

- ixnService.prometheus.podMonitor.enabled: `true`
- ixnVQNode.prometheus.podMonitor.enabled: `true`

For more information about modifying the Helm chart, see Configure Interaction Server.

Service	CRD or annotations?	Port	Endpoint/ Selector	Metrics update interval
Interaction Server (IXN)	PodMonitor	13131, 13133, 13139	option **ixnServer.ports.health** - default port 13131 - Endpoint: “/health/prometheus/all” option **ixnNode.ports.default** - default port 13133 - Endpoint: “/metrics” option **ixnVQNode.ports.health** - default port 13139 - Endpoint: “/metrics” **Note**: The above options are references to ports that match endpoints. Use these options to perform the associated query.	Default

Configure metrics

The metrics that are exposed by the Interaction Server service are available by default. No further configuration is required in order to define or expose these metrics. You cannot define your own custom metrics.

Alerting

No alerts are defined for Interaction Server.

Logging

Content coming soon

Interaction Server (IXN) metrics and alerts

Contents

- 1 Metrics
- 2 Alerts

Find the metrics Interaction Server (IXN) exposes and the alerts defined for Interaction Server (IXN).

Service	CRD or annotations?	Port	Endpoint/Selector	Metrics update interval
Interaction Server (IXN)	PodMonitor	13131, 13133, 13139	option **ixnServer.ports.health** - default port 13131 - Endpoint: “/health/prometheus/all” option **ixnNode.ports.default** - default port 13133 - Endpoint: “/metrics” option **ixnVQNode.ports.health** - default port 13139 - Endpoint: “/metrics” **Note**: The above options are references to ports that match endpoints. Use these options to perform the associated query.	Default

Metrics

This table includes IXN Server metrics and IXN Node metrics. IXN Node metrics begin with the prefix *ixnnode*.

Note: There are more metrics than the ones listed in the table. However, only the metrics listed in the table are supported.

Metric and description	Metric details	Indicator of
ixn_health_info_current_clients Indicates the number of clients that are connected to IXN at the moment.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:** 5	Workload
ixn_health_info_current_routers Indicates the number of 'connected to' IXN routers.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:** 1	Workload, Operability
ixn_health_info_client_count { client_type_name="Agent application" }	**Unit:** Amount **Type:** Gauge **Label:** client_type_name. See the metric description for more details.	Workload

Metric and description	Metric details	Indicator of

Indicates the number of clients with specified type, connected to IXN.
 	
Label descriptions:
 	
• client_type_name - type of connected clients. Possible values are:
  • Unknown
  • Proxy
  • Agent application
  • Media server
  • Reporting engine
  • Routing engine
  • Universal router
  • Third party client.
	Sample value: 101	

ixn_health_info_router_total_submitted { router_name="URServer" } Indicates the total number of interactions that have been submitted to the router. **Label descriptions**: router_name - the name of the router into which the interactions have been submitted.	**Unit:** Amount **Type:** Counter **Label:** router_name. See the metric description for more details. **Sample value:** 33	Workload, Operability
ixn_health_info_router_strategy_load_count { router_name="URServer", strategy_name="AAAstarterStrategy", strategy_tenant="1" } Indicates the number of strategies with specified name loaded into a specified router. **Label descriptions**: • router_name - the name of the router into which the interactions are loaded • strategy_name - name of strategy • strategy_tenant - tenant number	**Unit:** Amount **Type:** Gauge **Label:** router_name, strategy_name, strategy_tenant. See the metric description for more details. **Sample value:** 1	Workload

Metric and description	Metric details	Indicator of
ixn_health_info_router_current_capacity { router_name="URServer" } Indicates the current capacity of a specified router - the number of interactions, not including those already submitted, that can be submitted into the router. **Label descriptions**: router_name - name of router.	**Unit:** Amount **Type:** Gauge **Label:** router_name. See the metric description for more details. **Sample value:** 987	Workload, Operability
ixn_health_info_router_currently_submitted { router_name="URServer" } Indicates the number of interactions that are in a specified router. **Label descriptions**: router_name - name of router.	**Unit:** Amount **Type:** Gauge **Label:** router_name. See the metric description for more details. **Sample value:** 13	Workload, Operability
ixn_health_info_current_strategies Indicates number of strategies which are associated with active submitters.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:** 11	Workload
ixn_health_info_router_max_submitted { router_name="URServer" } Indicates the maximum capacity of specified router - the number of interactions, that can be submitted into the router. **Label descriptions**: • router_name - name of router. *ixn_health_info_router_max_submitted = ixn_health_info_router_currently_submitted + ixn_health_info_router_current_capacity*	**Unit:** Amount **Type:** Gauge **Label:** router_name. See the metric description for more details. **Sample value:** 1000	Workload, Operability
ixn_health_info_current_database_requests Indicates the current database requests queue length.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:** 0	Workload, Operability
ixn_health_info_total_database_requests Indicates the number of processed database requests from IXN application start till current moment.	**Unit:** Amount **Type:** Counter **Label:** None **Sample value:** 75	Workload, Operability
ixn_health_info_current_database_connections Indicates the current number of DB connections.	**Unit:** Amount **Type:** Gauge **Label:** None	Workload, Operability

Metric and description	Metric details	Indicator of
	Sample value: 5	
ixn_health_info_total_database_deadlocks Indicates the total number of database queries that end up with a deadlock for all the time since IXN started.	**Unit:** Amount **Type:** Counter **Label:** None **Sample value:** 0	Workload, Operability
ixn_health_info_router_strategy_last_submitted_at { router_name="URServer", strategy_name="AAAStarterToAgent", strategy_tenant="1" } Indicates the Unix timestamp when last interaction has been submitted to router for specified strategy. **Label descriptions**: • router_name - name of router; • strategy_name - name of strategy; • strategy_tenant - tenant number.	**Unit:** Unix timestamp **Type:** Gauge **Label:** router_name, strategy_name, strategy_tenant. See the metric description for more details. **Sample value:** 1618322383	Workload, Operability
ixn_health_info_queue_media_waiting_processing { queue_name="AAAstarterQueue", queue_tenant="1", media_name="chat" } Indicates the current number of interactions with specified media type that are waiting for processing in a specified queue. **Label descriptions**: • queue_name - name of queue; • queue_tenant - tenant number; • media_name - media type. Note: This value is provided in Pulse as well.	**Unit:** Amount **Type:** Gauge **Label:** queue_name, queue_tenant, media_name. See the metric description for more details. **Sample value:** 0	Workload
ixn_health_info_agent_logged_in_count { agent_tenant="1" } Indicates the current number of logged in agents. **Label descriptions**:	**Unit:** Amount **Type:** Gauge **Label:** agent_tenant. See the metric description for more details. **Sample value:** 565	Workload

Metric and description	Metric details	Indicator of
agent_tenant - tenant number.		
ixn_health_info_queue_media_in_router { queue_name="toAgent", queue_tenant="1", media_name="chat" } Indicates the number of the interactions with specified media type from a specified queue being routed. **Label descriptions**: • queue_name - name of queue • queue_tenant - tenant number • media_name - media type	**Unit:** Amount **Type:** Gauge **Label:** queue_name, queue_tenant, media_name. See the metric description for more details. **Sample value:** 10	Workload, Operability
ixn_health_info_queue_media_on_agent { queue_name="toAgent", queue_tenant="1", media_name="chat" } Indicates the number of the interactions with specified media type from specified queue being handled by agents. **Label descriptions**: • queue_name - name of queue • queue_tenant - tenant number • media_name - media type	**Unit:** Amount **Type:** Gauge **Label:** queue_name, queue_tenant, media_name. See the metric description for more details. **Sample value:** 5	Workload, Operability
ixn_health_info_queue_media_current_length { queue_name="toAgent", queue_tenant="1", media_name="chat" } Indicates the number of interactions with specified media type that are waiting processing in specified queue and were never delivered to agent. **Label descriptions**: • queue_name - name of queue • queue_tenant - tenant number • media_name - media type **Note**: This value is provided in Pulse as well.	**Unit:** Amount **Type:** Gauge **Label:** queue_name, queue_tenant, media_name. See the metric description for more details. **Sample value:** 2	Workload, Operability
ixn_health_info_queue_media_in_processing	**Unit:** Amount	Workload, Operability

Metric and description	Metric details	Indicator of
{ queue_name="toAgent", queue_tenant="1", media_name="chat" } Indicates the sum of the interactions with specified media type from specified queue being routed by routers and being handled by agents. **Label descriptions**: • queue_name - name of queue • queue_tenant - tenant number • media_name - media type *ixn_health_info_queue_media_in_processing = ixn_health_info_queue_media_in_router + ixn_health_info_queue_media_on_agent* **Note**: This value is provided in Pulse as well.	**Type:** Gauge **Label:** queue_name, queue_tenant, media_name. See the metric description for more details. **Sample value:** 15	
ixn_health_info_router_strategy_currently_submitted { router_name="URServer", strategy_name="AAAStarterToAgent", strategy_tenant="1" } Indicates the number of interactions which are submitted to specified router by specified strategy at the moment. **Label descriptions**: • router_name - name of router • strategy_name - name of strategy • strategy_tenant - tenant number	**Unit:** Amount **Type:** Gauge **Label:** router_name, strategy_name, strategy_tenant. See the metric description for more details. **Sample value:** 3	Workload, Operability
ixn_health_info_router_strategy_current_capacity { router_name="URServer", strategy_name="AAAStarterToAgent", strategy_tenant="1" } Indicates the number of interactions that can be submitted more to specified router by specified strategy. **Label descriptions**: • router_name - name of router • strategy_name - name of strategy • strategy_tenant - tenant number	**Unit:** Amount **Type:** Gauge **Label:** router_name, strategy_name, strategy_tenant. See the metric description for more details. **Sample value:** 197	Workload, Operability

Metric and description	Metric details	Indicator of
ixn_health_info_router_strategy_total_submitted { router_name="URServer", strategy_name="AAAStarterToAgent", strategy_tenant="1" } Indicates the number of interactions that were submitted to specified router by specified strategy since IXN app start till now. **Label descriptions**: • router_name - name of router • strategy_name - name of strategy • strategy_tenant - tenant number	**Unit:** Amount **Type:** Counter **Label:** router_name, strategy_name, strategy_tenant. See the metric description for more details. **Sample value:** 9	Workload, Operability
ixnnode_interactions_pulled_total Indicates the total number of the interactions pulled for the specific strategy. **Label descriptions**: strategy - The name of the strategy for which interactions are pulled.	**Unit:** Amount **Type:** Counter **Label:** strategy. See the metric description for more details. **Sample value:**	Workload, Operability
ixnnode_routing_sessions_current Indicates the current number of the routing sessions in routing.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:**	Workload
ixnnode_all_instructions_received_total Indicates the total number of instructions (of any type) received from ORS service.	**Unit:** Amount **Type:** Counter **Label:** None **Sample value:**	Workload, Operability
ixnnode_routing_instructions_received_total Indicates the total number of received routing instructions. **Label descriptions**: • strategy - The name of the strategy for which routing instructions is received. • type - The type of the instruction. It takes values "terminal" and "non-terminal". Terminal instructions are RequestDeliver, RequestPlaceInQueue,	**Unit:** Amount **Type:** Counter **Label:** strategy, type. See the metric description for more details. **Sample value:**	Workload, Operability

Metric and description	Metric details	Indicator of
RequestPlaceInWorkbin, RequestStopProcessing.		
ixnnode_redis_client_status Indicates the status of Redis client. **Label descriptions**: redis_client - The Redis client instance for which the metric is present. It takes values "reader" and "writer". Value: 0 - Not Ready 1 - Ready	**Unit:** Status **Type:** Gauge **Label:** redis_client. See the metric description for more details. er". **Sample value:**	Operability
ixnnode_redis_client_errors_total Indicates the total number of errors occurred on Redis client. **Label descriptions**: redis_client - The Redis client instance for which the metric is present. It takes values "reader" and "writer".	**Unit:** Amount **Type:** Counter **Label:** redis_client. See the metric description for more details. **Sample value:**	Error
ixnnode_redis_client_node_status Indicates the status of connection to individual nodes of Redis server (in singleton mode matches to ixnnode_redis_client_status). **Label descriptions**: • redis_client - The Redis client instance for which the metric is present. It takes values "reader" and "writer". • node - The node of Redis server for which the metric is present as "host:port". Value: 0 - Ready 1 - Not Ready 2 - Wait (so far there have been no connection attempts)	**Unit:** Status **Type:** Gauge **Label:** redis_client, node. See the metric description for more details. **Sample value:**	Operability
ixnnode_redis_client_node_errors_total Indicates the total number of errors occurred on individual nodes of Redis client (in singleton mode matches to	**Unit:** Amount **Type:** Counter **Label:** redis_client, node. See the metric description for more details.	Error

Metric and description	Metric details	Indicator of
ixnnode_redis_client_errors_total). **Label descriptions**: • redis_client - The Redis client instance for which the metric is present. It takes values "reader" and "writer". • node - The node of Redis server for which the metric is present as "host:port".	**Sample value:**	
ixnnode_redis_commands_completed_total Indicates the total number of successfully completed redis commands. **Label descriptions**: • redis_client - The Redis client instance for which the metric is present. It takes values "reader" and "writer". • command - The Redis command for which the metric is present. It takes values "xadd", "xread", "xdel".	**Unit:** Amount **Type:** Counter **Label:** redis_client, command. See the metric description for more details. **Sample value:**	Workload, Operability
ixnnode_redis_commands_failed_total Indicates the total number of failed redis commands. **Label desriptions**: • redis_client - The Redis client instance for which the metric is present. It takes values "reader" and "writer". • command - The Redis command for which the metric is present. It takes values "xadd", "xread", "xdel".	**Unit:** Amount **Type:** Counter **Label:** redis_client, command. See the metric description for more details. **Sample value:**	Error
ixnnode_rq_client_status Indicates the status of connection to RQ Service nodes. **Label descriptions**: rq_node - RQ Service node for which the metric is present.	**Unit:** Status **Type:** Gauge **Label:** rq_node. See the metric description for more details. **Sample value:**	Operability

Metric and description	Metric details	Indicator of
ixnnode_rq_requests_failed_total Indicates the total number of failed requests to RQ Service. **Label descriptions**: type - The type of the failed requests. It takes values "isp_event" - interaction protocol evnts and "ixn_ping" - health check messages.	**Unit:** Amount **Type:** Counter **Label:** type. See the metric description for more details. **Sample value:**	Error
ixnnode_instructions_processing_queue_concurrency_limit Indicates the maximum number of routing instructions that can be processed in parallel.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:**	n/a
ixnnode_instructions_processing_queue_size Indicates the number of instructions received from ORS currently being processed. **Label descriptions**: type - The type of the instruction. It takes values "isp_request" - routing instruction and "ixn_ping" - reply to health check message.	**Unit:** Amount **Type:** Gauge **Label:** type. See the metric description for more details. **Sample value:**	Workload, Operability
ixnnode_pull_request_total Indicates the total number of RequestPull requests successfully completed by InteractionServer. **Label descriptions**: strategy - The strategy for which interactions are pulled.	**Unit:** Amount **Type:** Counter **Label:** strategy. See the metric description for more details. **Sample value:**	Workload, Operability
ixnnode_route_request_sent_total Indicates the total number of route requests successfully sent to ORS. **Label descriptions**: strategy - The strategy to which requests are sent.	**Unit:** Amount **Type:** Counter **Label:** strategy. See the metric description for more details. **Sample value:**	Workload, Operability
ixnnode_route_request_failed_total Indicates the total number of route requests failed to send to ORS. **Label descriptions**: strategy - The strategy to which requests are sent.	**Unit:** Amount **Type:** Counter **Label:** strategy. See the metric description for more details. **Sample value:**	Error
ixnnode_instructions_processed_by_ixn_current	**Unit:** Amount	Workload

Metric and description	Metric details	Indicator of
Indicates the number of routing instructions currently being processed by IXN Server.	**Type:** Gauge **Label:** None **Sample value:**	
ixnnode_interactions_placed_back_total Indicates the total number of times an interaction was placed back in queue. **Label descriptions**: • reason - The reason of placing back in queue. It takes values: "StrategyOldQueueRequest" - Strategy explicitly requested to place to valid queue with name (not "BACK") matching the name of queue interaction was pulled from. Set by ORS. "StrategyBackRequest" - Strategy requested placing interaction "BACK" explicitly. Set by ORS. "Implicit" - Strategy did nothing and ORS places interaction back cause there are no instructions for it. Set by ORS. "Error" - ORS places interaction back into queue due to some error regardless of the error source be it strategy itself or any other reason. Set by ORS. "SubmitError" - IXN Node failed to send interaction to ORS and places it back into queue. "Unknown" - The reason was not specified by ORS. • strategy - The strategy which routed interactions.	**Unit:** Amount **Type:** Counter **Label:** reason, strategy. See the metric description for more details. **Sample value:**	Workload, Error, ORS Error
ixnnode_running_strategies_current Indicates the number of the strategies for which interactions currently are being pulled.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:**	Operability
ixnnode_configured_strategies_current Indicates the number of the strategies read from configuration for which interactions should be pulled.	**Unit:** Amount **Type:** Gauge **Label:** None **Sample value:**	Operability

Metric and description	Metric details	Indicator of
ixnnode_configuration_fetch_errors_total Indicates the total number of error occurred while fetching configuration from Configuration Service.	**Unit:** Amount **Type:** Counter **Label:** None **Sample value:**	Error
ixnnode_last_fetched_configuration_timestamp Indicates the last time the configuration was successfully fetched from Configuration Service as the number of seconds since January 1 1970 UTC.	**Unit:** Timestamp **Type:** Gauge **Label:** None **Sample value:**	Operability

Alerts

No alerts are defined for Interaction Server.

Logging

Learn how to store logs for Interaction Server.

Related documentation:

-
-
-

RSS:

- For private edition

IXN Node and IXN VQ Node containers log to stdout in structured JSON format.

IXN Server logging is configurable via configuration in ConfigServer. Refer to InteractionServer-log for more information.

IXN Server can be set up to output logs to stdout. Or it can be configured to write logs to disk. In the latter case, log volume should be able to write up to 100 MB/min and 10 MB/s for 2 minutes in peak.

Find the sample Helm values file. Keep in mind that you have to provide fluent-bit 1.8.x image from the vendor and keep it in the same registry as other images are pushed to.

```
ixnService:
 image:
   logSidecar:
     enabled: true
     repository: "fluent/fluent-bit"
     tag: "1.8.5"

 ixnServer:
   logStorage:
     mountPath: "/mnt/logs"
     storageSize: 1Gi
     storageClassName:
     volume:
       emptyDir: {}
```