WebRTC Private Edition Guide

Before you begin

8/13/2025

Contents

Find out what to do before deploying WebRTC.

Related documentation:

-
-
-

RSS:

- For private edition

Limitations and assumptions

All prerequisites described under Third-party prerequisites, Genesys dependencies, and Secrets have been met.

Download the Helm charts

Download the Helm charts from the webrtc folder in the JFrog repository. See Helm charts and containers for WebRTC for the Helm chart version you must download for your release.

For information about how to download the Helm charts in Jfrog Edge, see the suite-level documentation: Downloading your Genesys Multicloud CX containers

WebRTC contains the following containers:

Artifact	Type	Functionality	JFrog Containers and Helm charts
webrtc	webrtc gateway container	Handles agents' sessions, signalling, and media traffic. It also performs media transcoding.	https:////webrtc/webrtc/
coturn	coturn container	Utilizes TURN functionality	https:////webrtc/coturn/
webrtc-service	Helm chart		https://// webrtc-service-.tgz

Third-party prerequisites

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

The following are the third-party prerequisites for WebRTC:

Third-party services

Name	Version	Purpose	Notes
Keda	2.0	Custom metrics for scaling. Use of Keda or HPA is configurable through Helm charts.	KEDA can be enabled or disabled for WebRTC. But, WebRTC cannot be configured to use HPA instead of KEDA.
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.	
A container image registry and Helm chart repository		Used for downloading Genesys containers and Helm charts into the customer's repository to support a CI/CD pipeline. You can use any Docker OCI compliant registry.	
Command Line Interface		The command line interface tools to log in and work with the Kubernetes clusters.	

Storage requirements

WebRTC does not require persistent storage for any purposes except Gateway and CoTurn logs. The following table describes the storage requirements:

Persistent Volume	Size	Type	IOPS	Functionality	Container	Critical	Backup needed
webrtc-gateway-log-volume	50Gi	RW	medium	storing gateway log files	webrtc	Y	Y

webrtc-coturn-log-volume	50Gi	RW	medium	storing coturn log files	coturn	N	Y

Persistent Volume and Persistent Volume Claim will be created if they are configured. The size for them optional and should be adjusted according to log rate described below:

Gateway:

idle: 0.5 MB/hour per agent

active call: around 0.2MB per call per agent.

Example: For 24 full hours of work, where each agent call rate is constant and is around 7 to 10 calls per hour, we will require around ~500GB for 1000 agents, with around ~20GB being consumed per hour.

CoTurn:

For 1000 connected agents, the load rate is approximately 3.6 GB/hour which scales linearly and increases or decreases with the number of agents and stays constant whether calls are performed or not.

Network requirements

Ingress

WebRTC requires the following Ingress requirements:

- Persistent session stickiness based on cookie is mandatory. Stickiness cookie should contain the following attributes:
  - SameSite=None
  - Secure
  - Path=/
- No specific headers requirements
- Whitelisting (optional)
- TLS is mandatory

Secrets

WebRTC supports three types of secrets: CSI driver, Kubernetes secrets, and environment variables.

> **Important**
> GWS Secret for WebRTC should contain the following grants:

```
grant_type=authorization_code
grant_type=urn:ietf:params:oauth:grant-type:token-exchange
grant_type=urn:ietf:params:oauth:grant-type:jwt-bearer
grant_type=client_credentials
```

For GWS secrets, CSI or Kubernetes secret should contain `gwsClient` and `gwsSecret` key-values.

GWS secret for WebRTC must be created in the WebRTC namespace using the following specification as an example:

```
apiVersion: v1
kind: Secret
type: Opaque
metadata:
  name: webrtc-gws-secret
  namespace: webrtc
data:
  client_id: XXXXX
  client_secret: YYYYY
```

ConfigMaps

Not Applicable

WAF Rules

The following Web Application Firewall (WAF) rules should be disabled for WebRTC:

WAF Rule	Number of rules
REQUEST-920-PROTOCOL-ENFORCEMENT	920300
	920440
REQUEST-913-SCANNER-DETECTION	913100
	913101
REQUEST-921-PROTOCOL-ATTACK	921150
REQUEST-942-APPLICATION-ATTACK-SQLI	942430

Pod Security Policy

Not applicable

Auto-scaling

WebRTC and CoTurn auto-scaling is performed by KEDA operator. The auto-scaling feature requires Prometheus metrics. To know more about KEDA, visit https://keda.sh/docs/2.0/concepts/.

Use the following option in YAML values file to enable the deployment of auto-scaling objects:

```
deployment:
  keda:              true
```

You can configure the Polling interval and maximum number of replicas separately for Gateway pods and CoTurn pods using the following options:

```
gateway:
  scaling:
  pollingInterval:  30
  maxReplicaCount:  100

coturn:
  scaling:
    pollingInterval:  30
    maxReplicaCount:  100
```

- Gateway Pod Scaling
    - Sign-ins

```
gateway:
  scaling:
   pollingInterval:   30
   maxReplicaCount:   100
   prometheusAddress: http://monitoring-prometheus-prometheus.monitoring:9090
   thresholdSignins: 25
```

- CPU based scaling

WebRTC auto-scaling is also performed based on the CPU and memory usage. The following YAML shows how CPU and memory limits should be configured for Gateway pods in YAML values file:

```
gateway:
  scaling:
   prometheusAddress: http://monitoring-prometheus-prometheus.monitoring:9090
   pollingInterval:    30
   maxReplicaCount:    100
   thresholdSignins:   25
   thresholdCpu:       60
   thresholdMemory:    60
```

- CoTurn Pod scaling

Auto-scaling of CoTurn is performed based on CPU and memory usage only. The following YAML shows how CPU and memory limits should be configured for CoTurn pods in YAML values file:

```
coturn:
  scaling:
   pollingInterval:    30
   maxReplicaCount:    100
   thresholdCpu:       60
   thresholdMemory:    60
```

SMTP settings

Not applicable

Browser requirements

Browsers

Name	Version	Notes
Firefox	Current release or one version previous	Genesys also supports the current ESR release. Genesys supports the transitional ESR release only during the time period in which the new ESR release is tested and certified. For more information, see Firefox ESR release cycle. Firefox updates itself automatically. Versions of Firefox are only an issue if your IT department restricts automatic updates.
Chrome	Current release or one version previous	Chrome updates itself automatically. Versions of Chrome are only an issue if your IT department restricts automatic updates.
Microsoft Edge Chromium	Current release	

Genesys dependencies

WebRTC has dependencies on several other Genesys services and it is recommended that the provisioning and configuration of WebRTC be done after these services have been set up.

Service	Functionality

GWS	Used for environment and tenants configuration reading
GAuth	Used for WebRTC service and Agents authentication
GVP	Used for voice calls - conferences, recording, and so on
Voice microservice	Used to handle voice calls
Tenant microservice	Used to store tenant configuration

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

GDPR support

Not applicable