Tenant Service Private Edition Guide

Configure the Tenant Service

12/17/2025

Contents

Learn how to configure the Tenant Service.

Related documentation:

-
-
-
-
-

RSS:

- For private edition

Override Helm chart values

For additional information about overriding Helm chart values, see Overriding Helm Chart values in the *Genesys Multicloud CX Private Edition Guide*.

This section describes the purpose and use case for each configurable parameter in a Tenant Service deployment.

The content in the following tables is not intended to be actual values or the names of override options for the Helm charts; you can extract those later from the **values.yaml** file for each Helm chart.

Versioning

Group	Name	Purpose	Comments
version	Tenant image versions	Target image to install; must use same version for all init containers.	
version	Roles and permissions version	Target version of roles and permissions to apply.	
location	Image location	Target registry to pull images from.	

Identification

Group	Name	Purpose	Comments
Tenant	name	Nickname of the tenant.	Human-readable name. The default value is the Helm release name.

Group	Name	Purpose	Comments
Tenant	uuid	Unique identifier of all instances of the Tenant Service.	All nodes deployed to handle the end-customer environment use that UUID that is also registered. The last four positions of the UUID are used as the short Tenant ID, when applicable.

Backend parameters

Group	Name	Purpose	Comments
postgres	database host	A reference to the backend DBMS into which to persist the service.	Either a direct value or a file path that points to a mapped volume with file content to be used as the DBMS name.
postgres	database user	A reference to the backend database into which to persist the service.	Either a direct value or a file path that points to a mapped volume with file content to be used as the database username.
postgres	database name	A reference to the backend database into which to persist the service.	Either a direct value or a file path that points to a mapped volume with file content to be used as the database name.
postgres	database password parameters	Either direct value or reference to a secret name and key that hold a value; depends on relevant Kubernetes secret flag and Kubernetes secret env map flag.	These parameters control how password is being extracted by service. Direct value is supported to testing purposes. Secret name can be specified if password from external secret will be mounted as a volume and password exposed via file. Secret key can be specified if password will be mapped to environment variable.
postgres	Kubernetes secret usage flags	Indication of Kubernetes secret being used to keep password and whenever secret shall be mounted to env variable or expected to be mapped as a	Boolean values that force use of secret (instead of direct database password).

Group	Name	Purpose	Comments
		volume.	
postgres	ssl usage	specify secure connection preferences	Allow or require TLS.
consul	Kubernetes secret usage flags	indication of Kubernetes secret being used to keep token and whenever secret shall be mounted to env variable or expected to be mapped as a volume	boolean values that force use of secret (instead of direct consul token value).
consul	consul token parameters	either direct value or reference to a secret name and key that hold a value, depends on relevant Kubernetes secret flag and Kubernetes secret env map flag	these parameters control how token is being extracted by service. direct value is supported to testing purposes; secret name can be specified if token from external secret shall be mounted as a volume and token exposed via file; secret key can be specified if password shall be mapped to environment variable
redis	Kubernetes secret usage flags	indication of Kubernetes secrets being used to keep connection strings and whenever secrets shall be mounted to env variables or expected to be mapped as a volumes	boolean values that forces use of secret (instead of direct connection strings). applicable for all type of redis connections supported by tenant
redis	redis connection string for tenant stream	either direct value or reference to a secret name and key that hold a value, depends on relevant Kubernetes secret flag and Kubernetes secret env map flag	these parameters control how string is being extracted by service. direct value is supported for testing purposes; secret name can be specified if value from external secret shall be mounted as a volume and connection parameters exposed via file; secret key can be specified if string shall be mapped to environment variable
redis	redis connection string for config cache	either direct value or reference to a secret name and key that hold a value, depends on relevant Kubernetes	these parameters control how string is being extracted by service. direct value is supported for testing

Group	Name	Purpose	Comments
		secret flag and Kubernetes secret env map flag	purposes; secret name can be specified if value from external secret shall be mounted as a volume and connection parameters exposed via file; secret key can be specified if string shall be mapped to environment variable
redis	cluster flag	type of redis backend to connect	indicate whenever backend for redis is cluster or standalone server (same is used for all types of redis endpoints)
kafka	Kubernetes secret usage flags	indication of Kubernetes secrets being used to keep connection strings and whenever secrets shall be mounted to env variables or expected to be mapped as volumes	boolean values that force use of secret (instead of direct connection strings).
kafka	Kafka connection string	either direct value or reference to a secret name and key that hold a value, depends on relevant Kubernetes secret flag and Kubernetes secret env map flag	these parameters control how string is being extracted by service. direct value is supported for testing purposes; secret name can be specified if value from external secret shall be mounted as a volume and connection parameters exposed via file; secret key can be specified if string shall be mapped to environment variable

Kubernetes parameters

Group	name	purpose	comments
Auth and Autorization	Service Account	Specify non-default service account that shall be associated with all PODs of a tenant service deployment	PODs will be assigned default account if not specified. If required, a separate account will be created during tenant deployment and set to use by all PODs. This could be required if consul registration of tenant service relies on

Group	name	purpose	comments
			tenant's POD having Kubernetes accounts named after tenant service being registered. Service account is being created with name matching service name of as tenant (as visible in consul)
Auth and Authorization	security context	Adjust security context to run with random user	If random user is used to launch tenant containers, an adjustment to security context is needed for all tenant containers. Tenant is running as user 500 in group/ fsgroup 500 by default and it may use group/ fsgroup 0 for random user. For more information, see Security context configuration.
Auth and Autorization	Pod identity	specify optional annotation to associate with POD in order to access Kubernetes resources	PODs may be assigned an ADODB identity if needed
Scheduling	pod node selector	specify optional node pool selector for tenant PODs	PODs can be assigned to a specific pool, if needed. There is no default Kubernetes pool selections
Scheduling	pod toleration	specify optional toleration for tenant PODs	PODs can be set to tolerate specific taints, There is no default tolerations
Scheduling	affinity	enable affinity of tenant PODs to provide high availability	PODs of same tenant service may be forced to schedule in different locations (if supported by underlying infrastructure) using failure-domain.beta.kubernetes.io/ zone annotation when this is enabled. There is no affinity by default.
Scheduling	priority class	specify optional priority class for tenant PODs	PODS may be assigned specific priority class; not set by default.

Logging parameters

Group	name	purpose	comments
logging frameworks	fluentbit	Specify usage of fluentbit for tenant console logging	Tenant PODs won't use fluentbit logging solution by default. Tenant-specific fluentbit configuration can be enabled when deploying tenant-monitor resources, and use of fluentbit sidecar can be enabled on tenant PODs afterward.

logging frameworks	fluentbit config	Specify config map with fluentbit sidecar configuration that should be mounted as a volume.	If fluent is enabled in both tenant-monitor and tenant charts, a volume-referencing config map shall be specified, as below (if custom config map is created, map name can be set differently)
			`- name: tenants-fluent-bit-config`
			`configMap:`
			`name: tenants-fluent-bit-config`

logging frameworks	fluentbit local storage	Specify volume and volume mount where fluentbit logs will accumulate.	If fluent bit is enabled, the following volume is added:
			`- name: fluent-logs`
			`emptyDir: {}`
			and volume mount:
			`- name: fluent-logs`
			`mountPath: "/opt/genesys/logs/JSON"`

file logging	logging persistent volume	Specify usage of persistent volumes for logging by tenant components that produce log files.	Tenant PODs won't use persistent volume claims by default and no storage classes are created. Persistent

Group	name	purpose	comments
			volume claims can be added to tenant PODs when this is enabled.
file logging	logging persistent volume storage class	If use of persistent volume for logging has been enabled, storage class can be specified for these volumes as needed.	If tenant PODS are set to use persistent volume claims, storage class can be specified explicitly for volume claims. Storage class is not specified in PVC by default.
file logging	logging storage class	if logging to persistent volume is enabled and storage class has been specified explicitly then creation of storage class can also be enabled as needed	tenant-specific storage class for persistent volumes won't get created by default, can be enabled when deploying tenant-monitor resources and later utilized by tenant PODs using same storage class name. storage class name shall be specified for this to work
file logging	logging empty directory	default logging mount path pointing to a Kubernetes empty directory volume	part of default mounts provided for tenant service

Monitoring and observability

Group	name	purpose	comments
prometheus	Prometheus PodMonitor	Specify if PodMonitor to scrape Prometheus endpoints of tenant pods should be created.	Tenant-monitor doesn't create PodMonitor for Prometheus by default. Prometheus specific monitor can be created as part of infrastructure deployment when this is enabled.
prometheus	Pod level annotations	Alternative to enabling pod monitor.	

Integration

Group	name	purpose	comments
GWS	GWS endpoint	specify dedicated endpoint to register tenant service upon initial deployment or upgrade	when provided, define contact address to reach master GWS environment service that should be used to

Group	name	purpose	comments
			register tenant resources to make them available for GWS-based applications. by default, endpoint isn't specified and localhost connection attempt will be made by tenant container, GWS endpoint shall be reachable via service mesh upstream
GWS	GWS endpoint tls mode	enable https (secure) connection mode to contact GWS endpoint	when endpoint is provided, use of this parameter forces secure connection when accessing GWS on that address. NOTE: not available in initial release, only http connections are supported
GWS	Kubernetes secret usage flags	indication of Kubernetes secrets being used to keep client id and token and whenever secrets shall be mounted to env variables or expected to be mapped as volumes	boolean values that force use of secret (instead of direct client id and token).
GWS	client id and token	either direct values or references to secret names and keys that hold values, depends on relevant Kubernetes secret flag and Kubernetes secret env map flag	these parameters control how client id and token is being extracted by service. direct value is supported for testing purposes; secret name can be specified if value from external secret shall be mounted as a volume and connection parameters exposed via file; secret key can be specified if string shall be mapped to environment variables
Service Mesh	upstream services	override upstream service references used by tenant instance to locate intra-service and platform dependencies via consul	tenant POD has default service references that tenant service has to access via consul service mesh, this parameter allow to specify alternative values to make consul service names to tenant POD local ports

Group	name	purpose	comments
			allocated for service mesh.
voice	sip domain	provide sip communication domain for voice /sbc integration	customize SIP URI used to deliver between tenant and core voice platform and / or SBC
Tenant	service user	provide name of service user account	this account is being used to manage all parts of tenant service provisioning and is created at bootstrap
Tenant	Kubernetes secret usage flags	indicate usage of Kubernetes secrets to keep tenant service account password	
Tenant	service user password	either direct value or references to a secret name and key that hold values, depends on relevant Kubernetes secret flag and Kubernetes secret env map flag	these parameters control how password for internal admin account is being accessed service. direct value is supported for testing purposes; secret name can be specified if password from external secret shall be mounted as a volume and password exposed as a file; secret key can be specified if password shall be mapped to environment variables

Scalability and redundancy parameters

Group	name	purpose	comment
Tenant	node count	manage number of nodes deployed per service instance	by default service instance is being deployed with single node. if local high availability is required, additional nodes can be added to as service by value of this parameter and re-running deployment
Tenant	master location	manage location where master (writable) tenant node can be found for multi-regional deployments	by default tenant service is deployed as master, and expects to have writable local database backend accessible at its location, this parameter is required when

Group	name	purpose	comment
			deploying additional regions of the same tenant at other locations and its content should match with the name of consul datacenter where master tenant nodes are deployed. it shall be set the same across all locations

Extended parameters

Group	name	purpose	comment
postgres	main volume mounts for backend secrets	specify volumes and volume mounts for main tenant container that reference secrets to bound	usage of volumes and volume mounts depends on backend parameters selected as part of backend provisioning. volumes and mounts need to be specified if backend secrets are provisioned to come from secrets mapped as file systems. Note: Tenant has only one set of volume and volumeMounts entries in parameters overrides, all volumes for all purposes must be concatenated into one entry
postgres	init volume mounts for backend secrets	specify volumes and volume mounts for init containers (which are started as part of tenant onboarding and require secrets to operate)	Similar to volumes/ mounts of main tenant container. Note: Tenant has only one set of initVolumes and initVolumeMounts entries in parameters overrides, all volumes for all purposes must be concatenated into one entry.
consul	main volume mounts for consul token	specify volume mounts for main container to bound for consul access creds	usage of volumes and volume mounts depends on backend parameters selected as part of consul provisioning. Volumes and mounts need to be specified if consul token is being provisioned from secret mapped as

Group	name	purpose	comment
			file systems. Note: Tenant has only one set of volume and volumeMounts entries in parameters overrides, all volumes for all purposes must be concatenated into one entry
consul	init volume mounts for consul token	specify volume mounts for main container to bound for consul access creds	Similar to volumes/ mounts of main tenant container. Tenant has only one set of initVolumes and initVolumeMounts entries in parameters overrides, all volumes for all purposes must be concatenated into one entry.
redis			Only main container needs to be bound
kafka			Only main container needs to be bound

Configure Kubernetes

For information, see the following resources:

- Override Helm chart values
- Configure security
- Configure service-specific secrets
- Deploy Tenant Service

Configure security

Before you deploy the Tenant Service, be sure to read Security Settings in the *Setting up Genesys Multicloud CX Private Edition* guide.

Security context configuration

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

```
containerSecurityContext:
    # primaryApp containers' 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
    readOnlyRootFilesystem: false
    runAsNonRoot: true
    # base/centos7 uses uid=500
    runAsUser: 500
    runAsGroup: 500

  securityContext:
    # ref: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/#set-the-
security-context-for-a-pod
    # fsGroup is only valid at pod level
    fsGroup: 500
```

Arbitrary UIDs in OpenShift

If you want to use arbitrary UIDs in your OpenShift deployment, you must override the **securityContext** settings in the **values.yaml** file, so that you do not define any specific IDs.

```
containerSecurityContext:
    # primaryApp containers' 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
    readOnlyRootFilesystem: false
    runAsNonRoot: true
    # base/centos7 uses uid=500
    runAsUser: null
    runAsGroup: 0

securityContext:
    # ref: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/#set-the-
security-context-for-a-pod
    # fsGroup is only valid at pod level
    fsGroup: null
```

Configure service-specific secrets

Postgres database backend

Database backend can be allocated as shared or dedicated. The Tenant Service requires a separate database. Once deployed, secrets with details of Postgres backend parameters must be created as follows:

```
kubectl create secret generic dbserver -n voice --from-literal="dbserver="
kubectl create secret generic dbname -n voice --from-literal="dbname="
kubectl create secret generic dbuser -n voice --from-literal="dbuser="
kubectl create secret generic dbpassword -n voice --from-literal="dbpassword="
```

Service account password

The default account that allows access to the Tenant Service config interface after initial deployment can be supplied a password through a secret. If not provided, **password** will be used as a default value (empty passwords are prohibited by the Tenant Service).

```
kubectl create secret generic svcuseraccount -n voice --from-literal="svcpassword="
```

Genesys Authentication backend secrets

Genesys Web Services (GWS)/Genesys Authentication integration requires a client ID and token to allow the Tenant Service to register at GWS.

```
kubectl create secret generic gauthclientid -n voice --from-literal="clientid="

kubectl create secret generic gauthclientsecret -n voice --from-literal="clientsecret="
```