Genesys Authentication Private Edition Guide

Provision Genesys Authentication

5/10/2024

Contents

- Administrator

Learn how to provision Genesys Authentication.

Related documentation:

-
-
-

RSS:

- For private edition

> **Warning**
> Provisioning for Genesys Authentication is tied closely with other Genesys services. You must install these services before continuing with the steps on this page:
>
> - Tenant Service

Prerequisites

- You have installed the Genesys Authentication services and the following URLs are accessible:
  - /auth/v3/oauth/token
  - /environment/v3/environments
- You have the ops credentials (**services.secret.admin_username** and **services.secret.admin_password**) from the **values.yaml** file.
- The Tenant Service is accessible.
- You have Tenant Configuration Server service details such as hostname or IP, port, username, password, and cloud application name.

Create a new API Client

Make a POST request to create a new API client for Genesys Authentication:

```
curl --location --request POST '/auth/v3/ops/clients' \
--header 'Content-Type: application/json' \
--user ops:ops \ ---------- Cloud ops credentials () from values.yaml. The default value is
```

```
ops:ops
--data-raw '{"data": {
    "name": "external_api_client", ----------
    "clientType": "CONFIDENTIAL",
    "internalClient": true,
    "refreshTokenExpirationTimeout": 43200,
    "client_id": "external_api_client", ----------
    "client_secret": "", ----------
    "authorities": ["ROLE_INTERNAL_CLIENT"],
    "scope": ["*"],
    "authorizedGrantTypes": ["client_credentials", "authorization_code", "refresh_token",
"password"],
    "redirectURIs": ["https://gauth.","https://wwe.","https://gws.","https://prov."],
---------- should add gws/prov external URLS here
    "accessTokenExpirationTimeout": 43200
    }
}'
```

The result includes the **client_id** you need to Create an authentication token:

```
"status": {
        "code": 0
    },
    "data": {
        "clientType": "CONFIDENTIAL",
        "scope": [
            "*"
        ],
        "internalClient": true,
        "authorizedGrantTypes": [
            "refresh_token",
            "client_credentials",
            "password",
            "authorization_code",
            "urn:ietf:params:oauth:grant-type:token-exchange",
            "urn:ietf:params:oauth:grant-type:jwt-bearer"
        ],
        "authorities": [
            "ROLE_INTERNAL_CLIENT"
        ],

        "redirectURIs": [
            "https://gauth.",
            "https://gws.",
            "https://prov.",

        ],
        "accessTokenExpirationTimeout": 43200,
        "refreshTokenExpirationTimeout": 43200,
        "createdAt": 1619796576236,
        "name": "external_api_client",
        "client_id": "external_api_client",
        "client_secret": "secret",
        "encrypted_client_secret": "A34BOmXDedZwbTKrwmd4eA=="
    }
}
```

Create an authentication token

Make the following POST request to create an authentication token:

```
curl --location --user external_api_client:secret --request POST '/auth/v3/oauth/token' \
---------- user is the API client created in the previous step
--data-urlencode 'username=ops' \
--data-urlencode 'client_id=external_api_client' \ ---------- The client ID created in the
previous step
--data-urlencode 'grant_type=password' \
--data-urlencode 'password=ops'
```

The result includes the **access_token** you need to Add a Genesys tenant/environment:

```
{
    "access_token": "5f1ecb33-5c63-4606-8e30-824e494194c6",
    "token_type": "bearer",
    "refresh_token": "f0c7eed6-cc55-426f-9594-7ae14903e749",
    "expires_in": 43199,
    "scope": "*"
}
```

Add a Genesys tenant/environment

> ### Warning
> Complete this step after installing the Tenant service.

Make the following POST request to create the Environment tenant:

```
curl --location --request POST '/environment/v3/environments' \

--header 'Content-Type: application/json' \
--header 'Authorization: Bearer f3aa2109-8889-4182-b2b7-d86917c53e4e' \ ----- access token
generated in previous step
--data-raw '{
    "data": {
        "id" :  ---------- Tenant Service UUID you configured as part of the Tenant Service
deployment
        "username": "default", ---------- Tenant Configuration Server service username
        "password": "password", ---------- Tenant Configuration Server service password
        "connectionProtocol": "addp",
        "remoteTimeout": 7,
        "appName": "Cloud", ---------- Cloud app
        "traceMode": "CFGTMBoth",
        "tlsEnabled": false,
        "configServers": [{
            "primaryPort": 8888, ---------- Tenant Configuration Server service port
            "readOnly": false,
            "primaryAddress":
"tenant-839fa06b-6cbd-4af3-97d0-b579b850c4ec.voice.svc.cluster.local", ---------- Tenant
Configuration Server service URI
            "locations": "/USW1"

        }],
        "localTimeout": 5,
```

```
        "tenant": "Environment"
    }
}'
```

The result includes the environment ID you need to Add a contact center:

```
{
    "status": {
        "code": 0
    },
    "path": "/environments/d0fb6386-236c-4739-aec0-b9c1bd6173df" ---------- Environment ID
}
```

Add a contact center

> ### Warning
> Complete this step after installing the Tenant service.

Make the following POST request to add a contact center to the environment:

```
curl --location --request POST '/environment/v3/contact-centers' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer 9901f8d6-0351-47f8-b718-7db992f53a02' \
--data-raw '{
    "data": {

        "domains": [

        ],
        "environmentId": "343dd264-7c26-4f9e-82c5-26baedbcb797", ---------- Environment ID
created in the previous step
        "auth": "configServer",
        "id" :  ---------- Tenant Service UUID you configured as part of the Tenant Service
deployment
    }
}'
```

The result includes the contact center ID (also known as CCID) you will need to provision other Genesys services:

```
{
    "status": {
        "code": 0
    },
    "path": "/contact-centers/ed4c03f3-6275-4419-8b2b-11d14af10655" ---------- Contact center
ID
}
```

Add a data center

Make the following POST request to add a data center:

```
curl --location --request POST '/environment/v3/data-centers' \
--user ops:ops \
--header 'Content-Type: application/json' \
--data '{
    "data": {
        "location": "/usw1", ---------- The region as per Genesys Multicloud CX name
convention
        "entryPoint": , ---------- For the location above
        "readOnly": false/true ---------- This should be false for a primary or writeable
region only, true for all other regions
    }
}'
```

The result should look like this:

```
"status": {
        "code": 0
}
```

Update CORS settings

Make the following request to enable Cross-Origin Resource Sharing (CORS) and add URLs for each service that requires authentication. **Note:** CORS is required for Universal Contact Service.

Updates to the **value** field, which contains the list of URLs that require CORS permission, override any existing records. To preserve the existing records, make a GET request to collect the URLs and then append the new values as a comma separate list in your POST.

```
curl --location --request POST '/environment/v3/contact-centers//settings' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer 3f26790a-6e5b-4dc7-a139-ae78dab2a331' \ ---------- Bearer
token
--data-raw '
{
   "data":{
      "location":"/",
      "name":"cors-origins",
      "shared":"true",
      "value":",," ---------- URLs that require CORS permission
   }
}'
```

Clean up environments and contact centers

Complete the steps in this section to delete an environment and clean up its related objects, including the contact center and any related CORS settings.

Prerequisites

- You have created an API Client.
- The authentication token you generated previously.

Delete an environment

First, get your environments:

```
curl --location --request GET 'https:///environment/v3/environments'  --header
'Authorization: Bearer '
```

The response includes all environments and related information:


```
{
   "status":{
      "code":0
   },
   "data":{
      "genesysEnvironments":[
         {
            "id":"9350e2fc-a1dd-4c65-8d40-1f75a2e080dd",
            "tenant":"Environment",
            "appName":"Cloud",
            "username":"default",
            "password":"password",
            "connectionProtocol":"addp",
            "localTimeout":60,
            "remoteTimeout":90,
            "traceMode":"CFGTMBoth",
            "tlsEnabled":false,
            "configServers":[
               {
                  "primaryAddress":"tenant-9350e2fc-a1dd-4c65-8d40-1f75a2e080dd-
cfgmaster.service.dc1.consul",
                  "primaryPort":8888,
                  "readOnly":false,
                  "locations":"/USW1",
                  "readFromDb":false,
                  "useConfigExporter":false,
                  "initDb":false
               }
            ],
            "proxyPort":0
         },
         {
            "id":"6350e2fc-a1dd-4c65-8d40-1f75a2e080dd",
            "tenant":"Environment",
            "appName":"Cloud",
            "username":"default",
            "password":"password",
            "connectionProtocol":"addp",
            "localTimeout":5,
            "remoteTimeout":7,
            "traceMode":"CFGTMBoth",
            "tlsEnabled":false,
            "configServers":[
               {
                  "primaryAddress":"tenant-9350e2fc-
```

```
a1dd-4c65-8d40-1f75a2e080dd.voice.svc.gke2-useast1.gcpe002.gencpe.com",
                  "primaryPort":8888,
                  "readOnly":false,
                  "locations":"/USW1",
                  "readFromDb":false,
                  "useConfigExporter":false,
                  "initDb":false
               }
            ],
            "proxyPort":0
         }
      ]
   }
}
```

Make note of **genesysEnvironments.id** and use this ID to delete the environment:

```
curl --location --request DELETE 'https:///environment/v3/environments/' --header
'Authorization: Bearer '
```

The response:

```
{
   "status":{
      "code":0
   }
}
```

Repeat the delete request for each environment. Confirm all environments are deleted by calling GET /environment/v3/environments again.

Delete a contact center

First, get your contact centers:

```
curl --location --request GET 'https:///environment/v3/contact-centers/' --header
'Authorization: Bearer '
```

The response includes all contact centers and related information:

```
{
   "status":{
      "code":0
   },
   "data":{
      "contactCenters":[
         {
            "id":"9350e2fc-a1dd-4c65-8d40-1f75a2e080dd",
            "environmentId":"9350e2fc-a1dd-4c65-8d40-1f75a2e080dd",
            "domains":[
               "t100"
            ],
            "auth":"configServer"
         },
         {
            "id":"6350e2fc-a1dd-4c65-8d40-1f75a2e080dd",
            "environmentId":"6350e2fc-a1dd-4c65-8d40-1f75a2e080dd",
            "domains":[
```

```
               "t200"
            ],
            "auth":"config"
         }
      ]
   }
}
```

Make note of **contactCenters.id** and use this ID to delete the contact center:

```
curl --location --request DELETE 'https:///environment/v3/contact-centers/' --header
'Authorization: Bearer '
```

The response:

```
{
   "status":{
      "code":0
   }
}
```

Repeat the delete request for each contact center. Confirm all contact centers are deleted by calling GET /environment/v3/contact-centers/ again.