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.

Genesys Interaction Recording Solution Guide

Deploying Voice Processor

4/15/2025

Contents

Deploying Voice Processor

Genesys Interaction Recording (GIR) needs Voice Processor to process recording metadata from Media Control Platform (MCP), combine this metadata with data collected from Genesys Info Mart (GIM), and forward the result to Interaction Recording Web Services (RWS) and SpeechMiner Interaction Receiver (SM IR). During this process, recordings from multiple call legs are combined into a single interaction.

> **Important**
>
> If you are not using Voice Processor, Recording Processor Script (RPS) can be used. However, for new deployments, Genesys recommends using Voice Processor instead of RPS.

This topic contains the following sub-topics:

Comparison of Voice Processor and RPS

Voice Processor is a multi-threaded microservice based on the Node.js platform and it replaces the Python-based Recording Processor Script (RPS). The key advantage of Voice Processor is that since Node.js is a multi-threaded platform, a single Voice Processor instance can handle an incoming recording post load equivalent to 30-40 instances of RPS (20+ recordings per second). Therefore, a single instance should be sufficient for most customers. For customers with extremely high volumes, or who require redundancy, the Voice Processor can be run behind a load balancer similar to the existing RPS deployments.

Another benefit is that, in the event of outages that prevent posting of recordings to RWS and SpeechMiner Interaction Receiver, the Voice Processor automatically retries these posts for up to 40 days. As a result, you do not have to manually recover recordings if you resolve the downstream outage within that period.

The Voice Processor retrieves additional metadata from Genesys Info Mart (GIM) instead of Interaction Concentrator Database (ICON). The format and contents of the metadata posted by the Voice Processor to RWS and SM IR do not differ significantly from the format and contents posted by

RPS. However, there are some differences that may impact third-party integrations that download recording metadata from RWS, Recording Backup Service (RCBS), or from SpeechMiner. You must consider the following differences in the format and contents of the metadata:

- Name of the **eventID** property in the **eventData** list is different.
- Metadata that is meant to be internal-use only is not posted by the Voice Processor.
- As Genesys Info Mart is a data warehouse that is updated on a periodic basis through ICON data, the arrival of recordings in SpeechMiner is slightly delayed when compared to RPS.

Prerequisites

- Docker version 17.12.1-ce or higher running on a x86_64 Linux host.
- Ansible 2.6 or higher installed on the Docker host and the deployment is using Red Hat Enterprise Linux 7.
- Ansible 2.13 or higher installed on the Docker host and the deployment is using Red Hat Enterprise Linux 8.
- PostgreSQL 12.11 or higher.
- Genesys Info Mart 8.5 or higher installed on Microsoft SQL Server or PostgreSQL. For information on the system requirements for GIM, see Genesys Info Mart Requirements.
- Interaction Recording Web Services (RWS) 8.5.201.90 or higher.
- SpeechMiner 8.5 or higher if you are using SpeechMiner. We recommend that you install Speechminer before deploying Voice Processor.

We recommend that you have the following details before proceeding with deployment:

- Host name, port, database name for Genesys Info Mart, and user (read-only) credentials.
- Host name and port for the Interaction Recording Web Services (RWS), and Operation Admin (ops) credentials.
- Configuration Manager credentials for an account with access to the IVR Profile.
- Host name, port, and credentials needed to post to SpeechMiner Interaction Receiver. These details are required only for new installations.

Preparing your Docker environment

Extracting installation files

You can download the Docker image from the Genesys customer portal. The Docker image is a .tar file that contains the installation and configuration files required to set up and run the Voice Processor. Extract and copy the files from the image using the following steps:

1. Load the Docker image.

   ```
   zcat <.tar file> | docker load
   ```

2. View the list of Docker images and make a note of the newly loaded image.

   ```
   docker image ls
   ```

3. Add a custom tag to the docker image for your reference.

   ```
   docker tag <image ID> <tag>
   ```

4. Copy the files from the image.

   ```
   id=$(docker run --rm -dt <image> cat) && docker cp $id:/rps/compose . && docker stop $id
   ```

Now you have the sample configuration files in **./compose/defaults**, an Ansible playbook in **./compose** to help you set up and run the Voice Processor, and an SQL file for database setup. You will need these files for installing and configuring the Voice Processor.

> **Tip**
>
> You can refer to the custom tag of your docker image when setting parameters in the configuration files such as **settings-override.yml**, **secrets.yml**, and **docker-config.yml**.

Setting up Docker

Docker on the host must be running in swarm mode, but a multi-host swarm is not required. A new network inside Docker is used for the deployment. Docker swarm and its network can be set up this way:

```
docker swarm init
docker network create gir_vp --driver overlay --scope swarm
```

If the network that you created collides with an existing network in your environment, you can define the network's IP range like this:

```
docker network create --subnet 10.99.99.0/24 --gateway 10.99.99.1 --scope swarm gir_vp
```

Docker configs and secrets

Configs and secrets are Docker objects available in Docker swarm mode for storing run-time container configuration files and are mounted inside the container at run-time. The key difference between the two is that secrets are encrypted in Docker when at rest.

Docker logs

Docker container logs are typically stored under **/var/lib/docker/containers**, but container logs can be accessed simply by docker logs <container name>. Logs are rotated based on run-time configurations.

Configuring Voice Processor

This section contains the following sub-sections:

PostgreSQL database configuration

The Voice Processor requires a service-specific database that tracks work in progress items. This database runs on a PostgreSQL server. Set up the database using the following steps:

1. Create a database in your PostgreSQL server for the Voice Processor.
2. Create a PostgreSQL user and grant all privileges to the database that you created in the previous step.
3. Assign a password to the user that does not contain a backward slash (\) or quotation marks, as they might cause issues later.
4. Make a note of the database name, user name, and password — they are needed when configuring the Voice Processor in later steps.
5. Confirm that the **standard_conforming_strings** parameter of the PostgreSQL server is set to on (default).

> ### Important
>
> - GIR Voice Processor must have a separate PostgreSQL DB from Config Server since the Voice Processor PostgreSQL DB requires the `standard_conforming_strings` setting to be on and the Config Server PostgreSQL DB requires the `standard_conforming_strings` setting to be off.
> - Voice Processor supports connections to PostgreSQL DB when `password_encryption` is set to md5 or scram-sha-256 in the **postgresql.conf** file.
> - When using a PostgreSQL DB for Genesys Info Mart, if `password_encryption` is set to scram-sha-256 in the **postgresql.conf** file, the Genesys Info Mart version must be 8.5.016.04 or higher.

6. Run the provided script, `create_node_rps_tables_v2.sql`, against this new database to provision it.

> **Important**
>
> To avoid possible conflicts with their settings requirements, Genesys recommends not hosting the Voice Processor and Configuration Server databases on the same PostgreSQL instance.

Service level configuration

You can follow the instructions provided with the configuration files available in the default directory. You can copy the provided configuration files and make changes to your copies. We recommend that you use a version control repository to store your configurations. Add the PostgreSQL database, user name, and password to the **nodeRpsDb** setting in your copies of **settings-override.yml** and **secrets.yml**.

Voice Processor database settings

To enable TLS connection to the Voice Processor database, set the **ssl** parameter to `true` and configure the **trustedCA** parameter under **nodeRpsDb** in **settings-override.yml**.

```
nodeRpsDb:
  database: <database name>
  host: <db server hostname>
  port: <db port>
  user: <db user>
  ssl: < true / false >
  trustedCA: false / true / "<path to root certificate>"
```

The **ssl** parameter is optional and its default value is `false`. When you set it to `true`, the Voice Processor establishes a secure connection to the GIM database using TLS 1.2. Additionally, when the **ssl** parameter is set to `true`, the **trustedCA** parameter can be interpreted as follows:

- Do not authenticate the server certificate when the **trustedCA** value is `false`.
- Authenticate the server certificate against the system's root authorities when the **trustedCA** value is `true`.
- Authenticate the server certificate against the specified root authorities. Set **vpdb_ca_cert** in your copy of **docker-config.yml** with the `<path to root certificate>` value.

Voice Processor HTTPS settings

The **rwsBaseUri** setting in **settings-override.yml** supports HTTPS. For example:

```
https://<RWS hostname>:<RWS port>
```

To use HTTPS on the Voice Processor service API, set **https** to `true` in your copy of **docker-config.yml**. You must provide the server private key, public key, and path to the files.

```
https: true
tls:
  privkey: <path to the private key file>
  pubkey: <path to the public key file>
```

MCP post basic authentication

Add the following lines to the **settings-override.yml** file to enable basic authentication for the endpoint used by the MCP to post recording metadata:

```
authUsername: "<basic auth username>"
authPassword: "<basic auth password>"
```

If you add these options, you must also configure the Voice Platform profile option, **recording client.callrec_authorization**, in the **[gvp.service-parameters]** section to match these credentials. As basic authentication involves sending the credentials in plain text format, we strongly recommend that you use TLS for maximum security. Note that the other Voice Processor endpoints are not authenticated. Therefore, you must install the Voice Processor behind a firewall or API gateway to restrict access. You can obtain a summary of endpoints exposed by the Voice Processor service by accessing:

```
http://<GIR VP hostname>:<port>/apidoc
```

Setting the Voice Processor Docker image

Add the following line to the **docker-config.yml** file to specify the Voice Processor docker image that was imported:

```
image: <image ID>:<tag>
```

To find the values for `<image ID>:<tag>`, use the `docker images` command. An example is given below:

```
[root@CENTOS7-CloudVM defaults]# docker images
REPOSITORY                                          TAG                 IMAGE ID
CREATED             SIZE
voice-processor_v9000025                            latest              7320945e8b25
21 months ago       979MB
pureengage-docker-production.jfrog.io/gir_rp_nodejs   9.0.000.04.023      7320945e8b25
21 months ago       979MB
[root@CENTOS7-CloudVM defaults]#
```

Genesys Voice Platform profile configuration

Use HTTPS protocol in the Voice Processor URL when HTTPS is enabled in the Voice Processor service API.

```
recordingclient.callrec_dest = fixed,https://<VP hostname>:<VP port>/api/contact-
centers/<CCID>/recordings/
```

Use HTTPS protocol in the SpeechMiner Interaction Receiver URL when HTTPS is enabled on the SpeechMiner Interaction Receiver. When using HTTPS for the SpeechMiner URL, by default, the Voice Processor does not validate SpeechMiner server certificate. You can set **sm_ca_cert** in your copy of **docker-config.yml** with the `<path to root certificate>` value to authenticate the server certificate against the specified root authorities.

```
recordingclient.rp.speechminer_uri: fixed,https://<Speechminer backend
hostname>/interactionreceiver/
```

Tenant level configuration

As the Voice Processor is designed to support Genesys cloud multi-tenancy model, settings that may vary from tenant to tenant are stored in an RWS group settings called **rps-provisioning**:

> **Important**
>
> You need an Ops Admin user account to access these settings. For more information on how to update settings in RWS, see Settings API.

You must specify the Ops Admin user name and password in your copy of **secrets.yml**. The tenant level configuration values are set to the RWS group settings **rps-provisioning** using HTTP POST. For example:

```
curl -u <Ops admin user>:<password> -X POST -H "Content-Type: application/json"
<rwsBaseUri>/api/v2/ops/contact-centers/<ccid>/settings/rps-provisioning -d @rps-
settings.json
```

Where **rps-settings.json** contains settings like: `eventDataFilters`, `gimDb`, `rwsPostRecBaseUri` and others.

To confirm the Voice Processor per tenant settings, use HTTP GET. For example:

```
curl -u <Ops admin user>:<password> -X GET "<rwsBaseUri>/api/v2/ops/contact-
centers/<ccid>/settings/rps-provisioning?location=*&ignoreParentLocations=false"
```

GIM database

You must provide information needed to access the tenant's GIM database. To enable TLS connection to the GIM database, set the **ssl** parameter to `true` and configure the **trustedCA** parameter under GIM database settings in tenant level configuration.

```
{
    "name": "gimDb",
    "value": {
        "primary": {
            "host": "<GIM server hostname>",
            "port": <GIM server port (default 5432 for Postgres, 1433 for MS SQL)>,
            "user": "< DB user name >",
            "database": "<database name",
            "password": "<DB user password>",
            "dbType": "<postgres or mssql, default postgres>",
            "ssl": < true / false >,
            "trustedCA": false / true / "<path to root certificate>",
        },
        "backup": {
            < same settings as for primary >
        }
    }
}
```

The **ssl** parameter is optional and its default value is `false`. When you set it to `true`, the Voice Processor establishes a secure connection to the GIM database using TLS 1.2. Additionally, when the **ssl** parameter is set to `true`, the **trustedCA** parameter can be interpreted as follows:

- Do not authenticate the server certificate when the **trustedCA** value is `false`.
- Authenticate the server certificate against the system's root authorities when the **trustedCA** value is `true`
- Authenticate the server certificate against the specified root authorities by performing the following steps:
  1. Set **gim_ca_cert** in your copy of **docker-config.yml** with the <path to root certificate> value.
  2. Set **trustedCA** to `/rps/rpsdata/gimCA` in GIM database settings to be posted to tenant level configuration.

The **backup** parameter is optional. You can omit it if there is only one GIM database available.

RWS posting

You must specify the RWS instance to which recordings are posted. As this is a region-based setting, multi-regional deployments can ensure that recording data stays within the jurisdictional boundaries. The Voice Processor instance selects the location identified through the nodePath of the RWS server from which the setting is retrieved or the nearest matching parent. The **backup** parameter is optional. The URL supports HTTPS. When using HTTPS for the RWS URL, by default, the Voice Processor does not validate RWS server certificate. You can set **rws_ca_cert** in your copy of **docker-config.yml** with the <path to root certificate> value to authenticate the server certificate against the specified root authorities.

For example, the following setting applies to all Voice Processor instances:

```
{
    "name": "rwsPostRecBaseUri",
    "location": "/",
    "value": {
          "primary": "http://<hostname>:<port>{/<optional routing prefix>}",
          "backup": "http://<hostname>:<port>{/<optional routing prefix>}"
    }
}
```

The following setting would override the above global setting for Voice Processor instances that retrieved the setting from an RWS node with nodePath `/US` or `/US/*` :

```
{
    "name": "rwsPostRecBaseUri",
    "location": "/US",
    "value": {
          "primary": "http://<hostname>:<port>{/<optional routing prefix>}",
          "backup": "http://<hostname>:<port>{/<optional routing prefix>}"
    }
}
```

Event filtering

You can use filters to remove unwanted data from the recording metadata. The event filtering settings are similar to RPS except the mechanism of how the default filters are disabled.

```
{
    "name": "eventDataFilters",
    "value": {
        "attachedDataFilter": "regexp for new attached data filter",
        "attachedDataFilterException": "regexp for new attached data filter exception",
```

```
        "acwCustomDataFilter": "regexp for new ACW data filter",
        "acwCustomDataFilterException": "regexp for new ACW data filter exception"

        -- or, to disable the default filters or filter exceptions --

        "disableAttachedDataFilter": true,
        "disableAttachedDataFilterException": true,
        "disableAcwCustomDataFilter": true,
        "disableAcwCustomDataFilterException": true
    }
}
```

The default filters are:

- **attachedDataFilter**: `^ORSI:|^WWE|^PegAG`
- **attachedDataFilterException**: `^(GRECORD_(PARTITIONS|PROGRAM)|GSRS_STATE|GSIP_REC_FN)$`
- **acwCustomDataFilter**: `^ORSI:|^WWE|^PegAG`
- **acwCustomDataFilterException**: `^(GRECORD_(PARTITIONS|PROGRAM)|GSRS_STATE|GSIP_REC_FN)$`

Complete after-call work (ACW) threshold

The ACW threshold indicates how long the Voice Processor waits, in minutes, following the end of an interaction to update custom data. Custom data entered by agents after this interval is not added to recording metadata. The default value is zero.

```
{
    "name": "acwThresholdMinutes",
    "value": <ACW wait interval in minutes>
}
```

GIM DB ETL configuration

You must configure the GIM ETL application properly to ensure recording metadata is posted from the Voice Processor to SpeechMiner in a timely manner.

The **etl-start-time**, **etl-end-time**, and **etl-timezone** options in the **[schedule]** section are used to configure a daily maintenance period during which population of GIM data is paused for maintenance purpose. New recordings posted to the Voice Processor during this period are not processed and they are held temporarily in a database until the maintenance period finishes and the relevant GIM data becomes available. You must configure the **maintain-start-time** option such that the GIM ETL maintenance job begins and completes during the maintenance period.

The **etl-frequency** option in the **[schedule]** section is used to specify the cycle time of the GIM ETL jobs that populate the recording metadata used by the Voice Processor. We recommend that you use the default value of one minute. Note that any time longer than 3 minutes may cause subsequent delays in recording posts. If a longer **etl-frequency** setting is used, then the value of the Voice Processor service setting, **rpsInitialInteractionTimeout**, should be increased accordingly.

The **user-event-data-timeout** option in the **[gim-etl]** section is used to ensure that custom attached data entered during after-call work is captured. You can increase the default value of one hour if your agents will spend more than a few minutes in after-call work.

> **Important**
> Consult Genesys before setting non-default values for the following options.

The **max-call-duration**, **merge-failed-is-link-timeout**, and **extract-data-stuck-threshold** options in the **[gim-etl]** section must be configured properly to ensure completeness of the call metadata recorded in GIM. For more information on these options, see Operations-Related Options for Genesys Info Mart.

Deploying and Starting Voice Processor

Deploy Voice Processor to the newly configured Docker swarm using Ansible, referencing your copies of the default configuration files. This step also starts Voice Processor.

Before deploying, the **settings-override.yml** and **secrets.yml** files (or the yaml files you have designated to provide these settings) have several mandatory parameters that must be configured, as described below.

In the **settings-override.yml** file, the following parameters are required:

- **rwsBaseUri** – Specifies the address of the RWS cluster that will provide Voice Processor with contact center settings, including tenant-specific configurations such as Genesys Info Mart (GIM) database information and ACW Wait Time. Example: `http://some-rws-host.com:8090`
- **region** – Controls the `region` section in the metadata POSTed to RWS. This must match the **crRegion** setting of the RWS cluster to which recordings are posted. Example: `usa`
- **nodeRpsDb** – This section specifies the Voice Processor Persistence Database, which is required for storing recording metadata while Voice Processor is processing them.
  - **database** – Database on the host that will hold recording metadata. Example: `noderpsdb`
  - **host** – Host of the Persistence Database. Example: `noderpsdb.com`
  - **port** – Port that the Persistence Database is listening on. Example: `5432`

In the **secrets.yml** file, the following parameters are required:

- **nodeRpsDb** – This section specifies the credentials to the Voice Processor Persistence Database.
  - **user** – The user name to connect to the Persistence Database.
  - **password** – The password to connect to the Persistence Database.
- **rwsUserName** – The user name for authenticating with RWS.
- **rwsPassword** – The password for authenticating with RWS.

For an example of how the yaml files should be structured, you can refer to the default yaml files that were included with Voice Processor. These files are located at **<INSTALL_DIR>/defaults/**, where **<INSTALL_DIR>** is the location where you extracted the installation files to during the Preparing

your Docker environment step.

After configuring the default configuration files, deploy and start Voice Processor:

```
ansible-playbook \
       -e docker_config=<defaults directory of Your Docker Configuration Yaml (e.g. docker-
config.yml)> \
       -e logger_config=<defaults directory of Your Logger Configuration Yaml (e.g. logger-
config.yml)> \
       -e settings_override=<defaults directory of Your Voice Processor Configuration Yaml
(e.g. settings-override.yml)> \
       -e secrets=<defaults directory of Your Voice Processor Secrets Yaml (e.g.
secrets.yml)> \
        gir-vp-playbook.yml
```

> **Important**
>
> If the above options are not specified, then the .yml files in **./compose/defaults** will be used.

After starting the Voice Processor, update the Voice Processor endpoint (**/api/active-version**) with the version of your Voice Processor instance. You do not require any credentials to do this.

The setting to post the active version:

```
{ "version":"<GIR VP Version>" }
```

Example

```
curl -X POST -H "Content-Type: application/json" -d '\{ "version": "9.0.000.25" }'
girvp.company.com/api/active-version
```

Validating

1. Place a call to an agent or a test agent that is configured for recording.
2. Verify that the call arrives at the SpeechMiner UI. It should take 5 to 15 minutes depending on your configured ACW wait setting.
3. Assuming that live traffic is not recorded, you can use the health check endpoint `<domain:port>/api/status?verbose=1` . The items recordingsInProcess or the MCP Post operational status can be helpful in determining whether or not the recording is arriving at the Voice Processor. This also helps you to isolate GVP configuration problems from problems with the Voice Processor service. If a load balancer is used, the node serving the health check may not be the one that handled the recording. Therefore, several health checks may be required to cover the whole cluster.

Upgrading

Docker object configurations and secrets cannot be upgraded. We recommend that you remove the stack, update the required configurations, and redeploy.

```
docker stack rm <gir_vp>
ansible-playbook \
```

```
        -e docker_config=mydocker.yml \
        -e logger_config=mylogger.yml \
        -e settings_override=mysettings.yml \
        -e secrets=mysecrets.yml \
        gir-vp-playbook.yml
```

After starting the Voice Processor, update the Voice Processor endpoint (**/api/active-version**) with the version of your Voice Processor instance. You do not require any credentials to do this.

The setting to post the active version:

```
{ "version":"<GIR VP Version>" }
```

Example

```
curl -X POST -H "Content-Type: application/json" -d '\{ "version": "9.0.000.25" }'
girvp.company.com/api/active-version
```

Miscellaneous Docker tips

- To view network details:

  ```
  docker network inspect <network_name>
  ```

- To view a list of your swarm stacks:

  ```
  docker stack ls
  ```

- To view a list containers in your stack:

  ```
  docker stack ps <gir_vp>
  ```

- To view the container logs:

  ```
  docker logs <container name>
  ```

- To remove everything to start again:

  ```
  docker stack rm gir_vp
   docker network rm <network_name>
   docker swarm leave --force
  ```

Migrating from RPS to Voice Processor

This section explains how to migrate from an existing RPS deployment to Voice Processor.

Prerequisites

- Voice Processor is fully deployed

- The following Voice Processor dependent components are working as expected:
  - Interaction Recording Web Services (RWS)
  - Genesys Info Mart Database
  - SpeechMiner Interaction Receiver
  - Voice Processor Database

Migrating procedure

You can migrate from RPS to Voice Processor by changing the IVR profile, even if a Load Balancer is being used for RPS.

You must configure the **Recording Processor URI** parameter in the **Recording** tab of the IVR profile using Genesys Administrator Extension (GAX). This URI is used by Media Control Platform (MCP) to post metadata of the audio recording after the recording is complete. You must change this parameter to ensure that MCP posts metadata to the Voice Processor instead of RPS. For example:

```
http://<Voice Processor Host>:<Voice Processor Port>/api/contact-centers/<Contact Center
Domain Name>/recordings/
```

The value for the URI must always end with a forward slash (/). For more information, see Deploying Genesys Voice Platform for GIR.

> **Important**
>
> We recommend that you save the original value that is needed if rollback becomes necessary.

Validation

1. Place a test call.
2. After 15 to 20 minutes, check the SpeechMiner UI for the test call recording that should appear for the test agent.
3. RPS should no longer receive any call data.

Rollback

Restore the original value of the **Recording Processor URI** parameter in the IVR profile.

Shutting down RPS

Before shutting down RPS, recover any lost recordings after the RPS has processed existing calls. For more information, see Recovering Metadata for SpeechMiner. After ensuring that the Voice Processor is processing data as expected, shut RPS down. If there are any GIR ICONs used by RPS, shut them down as well.

Monitoring and troubleshooting information

The Voice Processor provides detailed health and performance information on the endpoint `<domain:port>/api/status` .

The following optional query parameters allow you to request specific health reports:

- `?verbose=1` provides a summary for each tenant and service.
- `?ccid=<HTCC ID>` provides a detailed report for a single tenant.
- `?service=<service name>` provides a detailed report for a single service. The following services are available for querying:
  - `persistence` provides information about the health and performance of the Voice Processor database.
  - `ccSettings` provides a status on the connection to RWS and the validity of the RWS settings.
  - `gim` provides health and performance information of the GIM database.
  - `rws` provides a status on RWS in the context of posting recording metadata.
  - `sm` provides health and performance information on data posted to SpeechMiner Interaction Receiver.
  - `schedRecovery` provides health and performance information on the internal scheduled recovery service which retries failed tasks periodically.
  - `mcpPosts` provides health and performance information on handling of incoming posts from MCP.

For example, if posts are not reaching SpeechMiner, a query to `/api/status?verbose=1` should provide sufficient information to isolate the problem. Additionally, you can provide a snapshot of the output from this query when you contact Genesys Customer Care for assistance.