# Genesys Interaction Recording Solution Guide

## Deploying Recording Processor Script

5/3/2025

# Deploying Recording Processor Script

## Recording Processor Script (Python 3)

### Important

Voice Processor, a multi-threaded microservice based on the Node.JS platform, is an alternative to the Recording Processor Script (RPS).

- For information on deploying Voice Processor, see Deploying Voice Processor.
- To migrate from an existing RPS deployment to Voice Processor, see Migrating from RPS to Voice Processor.
- RPS is not supported for deployments integrated with SIP Cluster. If your deployment uses SIP Cluster, you must use Voice Processor.

For new deployments, Genesys recommends using Voice Processor instead of RPS.

## Contents

## Prerequisites

Before installing and configuring the RPS, you must have the following prerequisites:

- An Interaction Recording Web Services 8.5.205.32 (or higher) instance where the call recording and screen recording metadata is stored.
- A Recording Crypto Server 8.5.095.16 (or higher) instance to decrypt the encrypted recordings.
- Network access to the WebDAV storage where the recordings are stored.
- For Recording Processor Script 8.5.500.13 (or higher), Recording Muxer Script must be upgraded to 8.5.500.10 (or higher).

## Installing Recording Processor Script

### Installing on Windows

1. Install 64 bit Python 3.11.5 from the Python website. To make Python 3 to work with OpenSSL 3.0.13, follow the below steps:
   - Download `libcrypto-3.dll` and `libssl-3.dll` from the Python Binary repository.
   - In `[python-source-folder]\DLLs`, replace with the above downloaded DLL files.
2. Install the **RPS IP** with the installer.

> **Important**
>
> Offline installation is supported starting from RPS 8.5.500.19.
>
> To install it in a fully offline environment, follow these steps:
>
> 1. Unzip the `<RPS>\thirdparty\flit_core-3.10.1.zip` file.
> 2. Run `py -m pip install . --no-build-isolation` from the `<RPS>\thirdparty\flit_core-3.10.1` directory.
> 3. Unzip the `<RPS>\thirdparty\wheel-0.45.0.zip` file.
> 4. Run `py -m pip install . --no-build-isolation` from the `<RPS>\thirdparty\wheel-0.45.0` directory.
>
> Also, add the flag `--no-build-isolation` for the upcoming commands when installing in an offline environment. For example, `- py -m pip install . --no-build-isolation`

> **Important**
>
> Install the following third-party libraries in the order they appear and unzip the files in `Administrator` mode.

3. Unzip the <RPS>\thirdparty\more-itertools-10.1.0.zip file.
4. Run `py -m pip install .` from the <RPS>\thirdparty\more-itertools-10.1.0 directory.
5. Unzip the <RPS>\thirdparty\jaraco.functools-4.0.0.zip file.
6. Run `py -m pip install .` from the <RPS>\thirdparty\jaraco.functools-4.0.0 directory.
7. Unzip the <RPS>\thirdparty\cheroot-10.0.0.zip file.
8. Run `py -m pip install .` from the <RPS>\thirdparty\cheroot-10.0.0 directory.
9. Unzip the <RPS>\thirdparty\web.py-0.62.zip file.
10. Run `py -m pip install .` from the <RPS>\thirdparty\web.py-0.62 directory.
11. Unzip the <RPS>\thirdparty\pyparsing-3.1.1.zip file.
12. Run `py -m pip install .` from the <RPS>\thirdparty\pyparsing-3.1.1 directory.
13. Unzip the <RPS>\thirdparty\httplib2-0.22.0.zip file.
14. Run `py -m pip install .` from the <RPS>\thirdparty\httplib2-0.22.0 directory.
15. Unzip the <RPS>\thirdparty\six-1.16.0.zip file.
16. Run `py -m pip install .` from the <RPS>\thirdparty\six-1.16.0 directory.
17. Unzip the <RPS>\thirdparty\python-dateutil-2.8.2.zip file.
18. Run `py -m pip install .` from the <RPS>\thirdparty\python-dateutil-2.8.2 directory.

## Installing on Linux (RHEL)

1. Install `zlib-devel (yum install zlib-devel)`.
2. Install `sqlite devel (yum install sqlite-devel.x86_64)`.
3. Install `libffi devel (yum install libffi-devel)`.
4. Install OpenSSL.
   - For 8.5.500.11 or lower versions, install OpenSSL version 1.1.1.
   - For 8.5.500.13 or higher versions, install OpenSSL 3.0.13. Download OpenSSL 3.0.13 from OpenSSL website and compile it. Example config command - `./config --prefix=/usr/home/openssl-3.0.13 --openssldir=/usr/home/openssl-3.0.13 --libdir=lib no-shared`
5. Install 64 bit Python 3.11.5.
   - For 8.5.500.11 or lower versions, compile with OpenSSL 1.1.1 from the Python website. While compiling Cpython 3.11.5 with custom openssl, use `--with-openssl` flag while compilation. Example config command - `./configure --with-openssl=/usr/home/openssl-1.1.1 --enable-optimizations`

- For 8.5.500.13 or higher versions, compile with OpenSSL 3.0.13 from the Python website.While compiling Cpython 3.11.5 with custom openssl, use `--with-openssl` flag while compilation. Example config command - `./configure --with-openssl=/usr/home/openssl-3.0.13 --enable-optimizations`

6. Install the **RPS IP** with the installer.

> **Important**
>
> Offline installation is supported starting from RPS 8.5.500.19.
>
> To install it in a fully offline environment, follow these steps:
>
> 1. Unzip the <RPS>\thirdparty\flit_core-3.10.1.zip file.
> 2. Run `py -m pip install . --no-build-isolation` from the <RPS>\thirdparty\flit_core-3.10.1 directory.
> 3. Unzip the <RPS>\thirdparty\wheel-0.45.0.zip file.
> 4. Run `py -m pip install . --no-build-isolation` from the <RPS>\thirdparty\wheel-0.45.0 directory.
>
> Also, add the flag `--no-build-isolation` for the upcoming commands when installing in an offline environment. For example, `- py -m pip install . --no-build-isolation`

> **Important**
>
> Install the following third-party libraries in the order they appear.

7. Untar the <RPS>/thirdparty/more-itertools-10.1.0.tar.gz file.
8. Run `python3 -m pip install .` from the <RPS>/thirdparty/more-itertools-10.1.0 directory.
9. Untar the <RPS>/thirdparty/jaraco.functools-4.0.0.tar.gz file.
10. Run `python3 -m pip install .` from the <RPS>/thirdparty/jaraco.functools-4.0.0 directory.
11. Untar the <RPS>/thirdparty/cheroot-10.0.0.tar.gz file.
12. Run `python3 -m pip install .` from the <RPS>/thirdparty/cheroot-10.0.0 directory.
13. Untar the <RPS>/thirdparty/web.py-0.62.tar.gz file.
14. Run `python3 -m pip install .` from the <RPS>/thirdparty/web.py-0.62 directory.
15. Untar the <RPS>/thirdparty/pyparsing-3.1.1.tar.gz file.
16. Run `python3 -m pip install .` from the <RPS>/thirdparty/pyparsing-3.1.1 directory.
17. Untar the <RPS>/thirdparty/httplib2-0.22.0.tar.gz file.
18. Run `python3 -m pip install .` from the <RPS>/thirdparty/httplib2-0.22.0 directory.
19. Untar the <RPS>/thirdparty/six-1.16.0.tar.gz file.

20. Run `python3 -m pip install .` from the <RPS>`/thirdparty/six-1.16.0` directory.
21. Untar the <RPS>`/thirdparty/python-dateutil-2.8.2.tar.gz` file.
22. Run `python3 -m pip install .` from the <RPS>`/thirdparty/python-dateutil-2.8.2` directory.

> **Important**
>
> - GIR does not support direct upgrade of RPS from Python 2 to Python 3.
> - Do not use the `setup.py install` command for installing libraries, instead use `pip install` command as mentioned above.

# Configuring Recording Processor Script

This section describes how to configure the Recording Processor Script for your environment.

## Configuring High Availability

### Recording Processor Cluster

RPS now provides High Availability support using multiple instances of RPS (all active). These active/active instances must be accessed through an HA proxy or load balancer. In this mode, each RPS is responsible for sending metadata to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner based on the load it receives. Each Recording Processor is responsible for fetching metadata from all ICON DB Servers. Media Control Platform (MCP) instances must be configured to access the Recording Processor instances by specifying a single URL that points to the load balancer.

To configure HA:

1. In each Recording Processor's **rpconfig.cfg** configuration file, in the **[processing]** section, set the following options:
   - **get_from_httc_before_posting** = 1
   - **mode** = `active`
2. Ensure that all Recording Processor instances have the *same* network related configuration.

> **Important**
>
> Genesys recommends that multiple Recording Processor instances be deployed on a single host to optimize the available CPU and take advantage of parallel processing. Multiple Recording Processor instances can then be deployed on other hosts as needed.

3. Using Genesys Administrator Extension, under the **Recording** tab in the IVR Profile, set the **Recording Processor URI** parameter to the load balancer's URL.
4. Configure the load balancer to balance traffic to the Recording Processor instances.

The following is an example configuration section that is needed for setting up an Apache load balancer for a three-instance Recording Processor cluster.

```
  ProxyPass /cluster balancer://nodecluster
    <Proxy balancer://nodecluster>
        BalancerMember http://<IP address of the active1 Recording Processor server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the active2 Recording Processor Server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the active3 Recording Processor Server>:<active
Recording Processor port>
    </Proxy>
```

> **Important**
>
> SpeechMiner version 8.5.2 or later is required for the Recording Processor cluster support to work properly.

### Recording Processor Script Active/Backup HA

RPS can also provide High Availability support by using two RPS instances (active and backup) accessed through an HA proxy or load balancer in failover mode. In this mode, the active RPS is always responsible for sending metadata to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner, and the backup instance is responsible for receiving and temporarily storing metadata if the active instance is unavailable. Once the active instance recovers, the balancer will direct clients to the active instance, and the backup instance will send any stored data to the active instance for metadata processing. Media Control Platform (MCP) instances must be configured to access the Recording Processor instances by specifying a single URL that points to the load balancer.

To configure HA:

1. In the active Recording Processor's **rpconfig.cfg** configuration file located in the **[processing]** section, set the **mode** parameter to `active`.
2. In the backup Recording Processor's **rpconfig.cfg** configuration file:
   - Set the **mode** parameter to `backup`.
   - In the **[processing]** section, set the **post_uri** parameter to `http://<active_rp_ip>:<active_rp_port>/api/contact-centers/%s/recordings/`.
3. Using Genesys Administrator Extension, under the **Recording** tab in the IVR Profile, set the Recording Processor URI parameter to the failover load balancer's URL.
4. Configure the load balancer to direct traffic to the active Recording Processor instance first and to the backup instance if an error/failure occurs.

The following is an example configuration section that is needed for setting up an Apache load

balancer in failover mode for Recording HA support.

```
   ProxyPass /cluster balancer://nodecluster
    <Proxy balancer://nodecluster>
        BalancerMember http://<IP address of the active Recording Processor server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the backup Recording Processor Server>:<active
Recording Processor port> status=H
    </Proxy>
```

For more information about how to use Genesys Administrator Extension to configure your Contact Center, see the Genesys Administrator Extension Help.

## Configure Passwords

> ### Important
>
> In a Linux or Windows environment, RPS supports reading the environment variables for password related configuration parameters in order to avoid storing the password in plain-text in the configuration file. When both are available, the environment variables take precedence.
>
> The following definitions describe the mapping of the environment variables to the corresponding configuration parameter:
>
> **HTCC_PASSWORD** – maps to the existing configuration parameter under the **htcc** section, password value.
> **AUTH_PASSWORD** - maps to the existing configuration parameter under the **auth** section, password value.
> **CONFIG_SERVER_PASSWORD** - maps to the existing configuration parameter under the **config_server** section, password value.
> **<ICON_ID>_DB_INFO_PASSWORD** - maps to the existing configuration parameter under the **<ICON_ID>_db_info** section, password value, where <ICON_ID> refers to the ICON instance listed in the **icon_db_servers** section.
>
> For example, if you have `VCCSIPSwitch: icon1` the environment variable that corresponds to the `icon1_db_info` password is `icon1_DB_INFO_PASSWORD`.

In a Windows environment only, the Recording Processor Script (RPS) can store passwords in the Windows Vault instead of in the **rpconfig.cfg** file or requiring the use of environment variables.

For example, run the following command for the Recording Processor Script credentials located at `<Recording Processor Directory>\rp`. This command will prompt the user to enter valid values for the password/key configuration parameters and stores the passwords in the encrypted file named **rp.secret**:

### Command to store:

```
encryptPassword.bat -password <password_string>
```

Where `<password_string>` is a comma-delimited series of key/value pairs, use the format `<environment variable name 1>=<environment variable value 1>,<environment variable name 2>=<environment variable value 2>,<environment variable name 3>=<environment variable value 3>`, and so on. Note that space is not allowed in `<password_string>`.

For example:

```
encryptPassword.bat -password "HTCC_PASSWORD=somepassword1, AUTH_PASSWORD=somepassword2,
CONFIG_SERVER_PASSWORD=somepassword3, ICON1_DB_INFO_PASSWORD=somepassword4,
ICON2_DB_INFO_PASSWORD=somepassword5"
```

> **Important**
>
> Passwords used with this command cannot contain a comma or an equals sign.

## Configure the Configuration Server Connection

To configure the Configuration Server connection, set the following parameters in the **[config_server]** section of the Recording Processor **rpconfig.cfg** configuration file:

| Parameter Name | Default Value | Description |
|---|---|---|
| application_name | Empty | Specifies the name of the RPS application object in the Configuration Server, when using RPS as a third party server application. |
| hostname | `<ip address>` | Specifies the IP address of the primary Configuration Server. |
| port | `2020` | Specifies the port of the primary Configuration Server. |
| username | `default` | Specifies the Configuration Server username. |
| password | `password` | Specifies the Configuration Server password.<br><br>**Note:**<br>The password can be overridden by the **CONFIG_SERVER_PASSWORD** environment variable. |
| backup_host | Empty | Specifies the IP address of the backup Configuration Server. |
| backup_port | Empty | Specifies the backup port of the backup Configuration Server. |

> **Important**
>
> Recording Processor Script does not support a secure connection to the Configuration Server.

## Configuring the Server Port

In the **[rp_server]** section of the **rpconfig.cfg** file, set the **port** parameter.

> **Important**
>
> You can also set the "port" parameter using the command line with the --port command line argument. The command line argument takes precedence over the configuration file value.

## Configuring the Connection to Interaction Recording Web Services (Web Services)

To configure the Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) connection, set the following parameters in the **[htcc]** section of the Recording Processor **rpconfig.cfg** configuration file:

| Parameter Name | Default Value | Description |
|---|---|---|
| base_uri | `http://<Web Services IP>:<Web Services Port>` | Specifies the Base URI for accessing the Interaction Recording Web Services (Web Services) API. |
| username | ops | Specifies the username used to access the Interaction Recording Web Services (Web Services) account. |
| password | ops | Specifies the password used to access the Interaction Recording Web Services (Web Services) account.<br><br>**Note:** The password can be overridden by the **HTCC_PASSWORD** environment variable. |

Each Interaction Recording Web Services (Web Services) instance must have a region associated with it. Set the region parameter in the [metadata] section of the rpconfig.cfg file to match the region associated with Interaction Recording Web Services (Web Services) instance set to receive the Recording Processor's metadata.

## Configuring Cross-Site Request Forgery (CSRF) Protection

If Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) has CSRF enabled, set the following parameter in the **[htcc]** section of the **rpconfig.cfg** file:

- **csrfp** = 1

## Configuring the Connection to SpeechMiner

To configure the SpeechMiner Connection:

1. In the IVR Profile, set the recording destinations to point to the SpeechMiner interaction receiver:
   a. Login to Genesys Administrator Extension, and navigate to **Configuration > System > Configuration Manager**.
   b. Under **Voice Platform**, select **Voice Platform Profiles**.
   c. Click on the IVR Profile for which you want to set the recording destination.
   d. Select the **Recording** tab.
   e. In the**SpeechMiner Interaction Receiver** field, enter the URL that points to the SpeechMiner service responsible for accepting metadata from the RPS for this profile. For example, `https://<SpeechMiner IP>/interactionreceiver`.
   f. In the **SpeechMiner Interaction Receiver Authorization Header** field, enter the authorization information (username:password) required to connect to the SpeechMiner service used by the RPS. For example, `user:password`.

> **Important**
> The values of these options must match the corresponding configuration options in the SpeechMiner system.

## Configuring Failed Message Files

The Recording processor can backup messages that fail to POST correctly to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner. These files are located in the **<recording processor dir>\failed** folder.

In the **rpconfig.cfg** configuration file, add the following parameter:

```
[processing]
backup_failed_metadata = 1
```

## Configuring the Agent Hierarchies

Recording Processor Script uses the agent hierarchy information to set the access control information for recordings within the recording metadata. Refer to Access Control for Genesys Interaction Recording Users to configure this appropriately.

## Configuring Basic Authorization

In the **rpconfig.cfg** configuration file, set the following parameters:

```
[auth]
# Basic Authentication username and password. Set username blank to disable.
username = rp_username
password = rp_password
```

### Important

- The username and password must match the username and password entered in the IVR Profile. For more information about configuring the IVR Profile, see the IVR Profile section.
- The password can be overridden by the **AUTH_PASSWORD** environment variable.

## Configuring After Call Work

Recording Processor can collect After Call Work (ACW) customized data from ICON.

In the **rpconfig.cfg** file, in the **[processing]** section, add the following parameters:

- **enable_acw**—Set it to 1.
- **acw_threshold_minutes**—Set it to the maximum time to wait for the customized attached data.

### Important

- If Call Customized Attached Data is still not available in the ICON database after *acw_threshold_minutes*, the RPS will stop collecting customized data for this recording and write it to the database.
- If *enable_acw* is set to `0`, ACW customized data will not be included.
- If disposition code is required in the metadata, you must set **enable_acw** and **acw_threshold_minutes** using Recording Processor configuration.
  The disposition code is part of the user data collected during ACW. For this reason, **enable_acw** must be enabled in the Recording Processor. If it is not enabled, the data will not be collected.
  If the disposition code must be collected from the Recording Processor, configure the following to include the disposition code for recording:
  `[processing] enable_acw=1`
  `[metadata] acw_threshold_minutes=5`. Where 5 is the maximum time (in minutes) to wait for the disposition code.
  In the ICON configuration, the **EventData** parameter in the **custom-states** section, must include `char,DispositionCode` and **store-event-data** must be set to `conf` to collect the attached data:
  `[custom-states] store-event-data=conf EventData=char,DispositionCode`
  For additional information, refer to the ICON Deployment Guide.

## Configuring ICON for Recording Processor

> **Important**
>
> When configuring Recording Processor to connect to a primary and backup ICON Database in HA mode, two separate DB Servers must be used. The DB Servers must run in an active/active pair mode.

To configure ICON, edit the **rpconfig.cfg** configuration file as follows:

1. Configure the switches:

   Add a configuration option for each switch name under the **[icon_db_servers]** section. You can specify more than two ICON databases per SIP Switch configuration. For example:

   ```
   [icon_db_servers]
   SIP_Switch1: icon1
   SIP_Switch2: icon2, icon2Backup
   SIP_Switch3: icon3, icon4, icon5, icon6
   ```

   > **Important**
   >
   > In the above example, **SIP_Switch3** has 4 ICON databases. The Recording Processor Script (RPS) keeps track of the ICON database instance currently used. If the current database instance becomes unavailable, RPS will attempt the operation in the next database.

   The configuration option name must match the exact name of the switch as configured in the Genesys configuration. The primary and backup ICON names must be unique, but do not have to match anything in the Genesys configuration.

2. Configure the ICON Connection Settings:

   - For each unique ICON specified in the first step, create a new section using the following syntax: `<ICON_ID>_db_info`, where `<ICON_ID>` corresponds to the values defined in the **[icon_db_servers]** section above.
   - **dbengine** must be `mssql`, `oracle`, `db2`, or `postgres`.
   - **dbserver_host** and **dbserver_port** specify the host and port information for the Genesys DB Server.
   - **dbms** specifies the host where the database resides.

   The following is an example using the values for **SIP_Switch1** and **SIP_Switch2** from step 1:

   ```
   [icon1_db_info]
   dbserver_host = vm221.us.int.genesyslab.com
   dbserver_port = 12201
   ```

```
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon2_db_info]
dbserver_host = vm222.us.int.genesyslab.com
dbserver_port = 12201
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon2Backup_db_info]
dbserver_host = vm223.us.int.genesyslab.com
dbserver_port = 12201
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon_oracle_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname =
dbms = <database host/Oracle SID>
dbengine = oracle

[icon_postgres_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname = <database name>
dbms = <database host>
dbengine = postgres

[icon_db2_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname =
dbms = <database host>
dbengine = db2
```

## Important

- For Oracle or DB2 implementations, the **dbname** parameter must be left blank or empty.
- The password can be overridden by the **<ICON_ID>_DB_INFO_PASSWORD** environment variable.

In the example above, the RPS will use the connection properties in section **[icon1_db_info]** when processing recording metadata from an MCP provisioned to `SIP_Switch1`. The RPS will use the connection properties in section **[icon2_db_info]** when processing recording metadata from an MCP provisioned to `SIP_Switch2`. In the case of `SIP_Switch2`, the RPS will use the connection settings in **[icon2Backup_db_info]** if the primary ICON (icon2) is unavailable when recording metadata is being processed.

## Configure how to Filter Metadata from ICON

The Recording Processor supports the ability to filter specific attached data fields (based on the key name), such as attached data and After Call Work (ACW) customized data retrieved from the ICON database. This support prevents specific metadata from reaching additional GIR related components (for example, SpeechMiner).

The following two sections describe how to:

- Filter attached data.
- Filter ACW.

### Important

- Verify that the following items are not removed from the filter. Removing these items may cause errors in GIR:
  - GRECORD_PARTITIONS
  - GRECORD_PROGRAM
  - GSIP_REC_FN
- When running SpeechMiner, you must include Workspace Web Edition (WWE) in the attached `attached_data_filter` and `acw_custom_data_filter` Recording Processor configuration values. For example:

```
[filter]
attached_data_filter=^ORSI:|^WWE
acw_custom_data_filter=^ORSI:|^WWE
```

### Filter Attached Data

1. Edit the **rpconfig.cfg** file.
2. Locate the **Filter** section. If the **Filter** section does not exist, add it as follows:

```
...
[filter]
...
```

3. Add a new option called **attached_data_filter** to the **Filter** section as follows. The value must be a Regex pattern used to filter out attached data whose key name matches the pattern.

```
...
[filter]
attached_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/3.11/library/re.html.

4. Add a new option called **attached_data_filter_exception** to this section as follows. The value must be a Regex pattern used to exclude key names that should not be filtered out (for example, like GRECORD_PARTITIONS).

```
...
[filter]
attached_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
attached_data_filter_exception = ^GRECORD_PARTITIONS$     ;   (Note: this is the default
value when the option is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/3.11/library/re.html.

5. Restart the Recording Processor.

## Filter ACW Related Custom Data

1. Edit the **rpconfig.cfg** file.
2. Locate the **Filter** section. If the **Filter** section does not exist, add it as follows:

```
...
[filter]
...
```

3. Add a new option called **acw_custom_data_filter** to the **Filter** section as follows. The value must be a Regex pattern used to filter out ACW whose key name matches the pattern.

```
...
[filter]
acw_custom_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/3.11/library/re.html.

4. Add a new option called **acw_custom_data_filter_exception** to this section as follows. The value must be a Regex pattern used to exclude ACW that should not be filtered out (for example, like GRECORD_PARTITIONS).

```
...
[filter]
acw_custom_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
acw_custom_data_filter_exception = ^GRECORD_PARTITIONS$     ;   (Note: this is the
default value when the option is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here:

https://docs.python.org/3.11/library/re.html.

5. Restart the Recording Processor.

## Configuring SSL for Recording Processor

To configure SSL:

### Configure HTTPS on the Primary Recording Processor Server

1. Create a self-signed certificate and private key for the Recording Processor host. For example, on RHEL run: `openssl req -new -x509 -days 1024 -nodes -out cert228.pem -keyout cert228key.pem`
2. In the `rp_server` section of the Recording Processor's configuration file, set the following parameters:
   - `ssl_certificate`—To point to the certificate PEM file. For example, `ssl_certificate=cert228.pem`.
   - `ssl_private_key`—To point to the private key file. For example, `cert228.pem`.
3. Give the self-signed certificate PEM file to any MCP client that needs to validate the certificate during the SSL handshake. See the "Enable Secure Communication" section Genesys Voice Platform 8.5 User's Guide.
4. Restart Recording Processor.

### Configure the HTTPS connection to Interaction Recording Web Services (Web Services)

1. Set up HTTPS on Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier). See the Genesys Security Deployment Guide.
2. Get the corresponding certificate for the Interaction Recording Web Services (Web Services) server. Set the caCertificate option in your Interaction Recording Web Services application (see caCertificate if you're using a Web Services application).
3. In the **[htcc]** section of the Recording Processor configuration file, set **base_uri** parameter to use `https`.
4. In the **[client]** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA certificate file.

> **Important**
>
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

### Configure the HTTPS connection to SpeechMiner

1. Set up HTTPS on SpeechMiner.See the Genesys Security Deployment Guide.
2. Set the **disable_ssl_certificate_validation** parameter in the **[speechminer]** section of the Recording Processor configuration to a value of 1.

3. Using Genesys Administrator Extension on the `Recording` tab of the IVR Profile, modify the `SpeechMiner Interaction Receiver` field use `https` as the protocol in the URL.
4. In the **[client]** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA cert file.

> **Important**
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

### Configure the HTTPS connection from the backup Recording Processor to the primary Recording Processor

1. Configure HTTPS on the primary Recording Processor.
2. Get the corresponding PEM certificate for the Web Services server.
3. In the **[processing]** section of the Recording Processor configuration file, set the **post_uri** parameter to use `https` as the protocol in the URL.
4. In the **client** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA certificate file.

> **Important**
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

## Configure the HTTPS on the backup Recording Processor Server

Follow the same procedure used for the Primary Recording Processor Server using a new certificate and private key for the Backup Recording Processor's server.

## Configuring the IVR Profile

Using Genesys Administrator Extension, configure the following parameters on the **Recording** tab of the IVR Profile:

1. **Recording Processor URI**—The URI that the Media Control Platform (MCP) uses to post the metadata of the audio recording after the recording is complete. For example, `http:// <Recording Processor Host>/api/contact-centers/<Contact Cente Domain Name>/recordings/`.

> **Important**
> The value for the URI must always end with a forward slash (/).

2. SpeechMiner Interaction Receiver—The URL that points to the SpeechMiner service responsible for accepting metadata from the RPS for this profile.
3. SpeechMiner Interaction Receiver Authorization Header—The authorization information required to connect to the SpeechMiner service used by the RPS. For example, `<SpeechMiner Webserver Username>:<SpeechMiner Webserver Password>`.

For more information, see the Configuring GVP.

## Configuring the Recording Processor Using Genesys Administrator Extension (Optional)

The Recording Processor uses a configuration file instead of a specific application object in Configuration Server. However, it is possible to configure the Recording Processor as a "third-party server" application enabling Genesys Administrator Extension to monitor, start, and stop the process.

The following steps describe how to setup Recording Processor as a "third party server" application in Genesys Administrator Extension. For more information, see the "Using the Management Layer" section of the Framework 8.5.1 Management Layer User's Guide

Configuring RPS to Start/Stop via LCA using Genesys Administrator Extension:

1. Install and deploy the latest RPS.
2. Make sure that the Local Control Agent (LCA) is running.
3. Create a new application template in Genesys Administrator Extension called `Recording Processor Script` of type `Third Party Server`.
4. Create a new application (for example, `myRPS`) in Genesys Administrator Extension using this new application template.
5. Set the `Command Line` parameter (for example, C:\Python311\python.exe).
6. Set the `Host` parameter in the application's server info to the correct Host object.
7. Set the `Working Directory` parameter to the `<Recording Processor Install Directory>\rp` directory. For example, `/opt/genesys/Recording_Processor_Script_8.5/rp/`.
8. Set the `Command Line Arguments` parameter to the appropriate values. For example, `recording_process.py --config-file=/opt/genesys/Recording_Processor_Script_8.5/rp/rpconfig.cfg`.
   Refer to the Starting the Recording Processor Script section for additional command line parameters
9. Make sure that LCA has permission to read and write to the Recording Processor installation directory and Recording Processor log directory.
10. Save the configuration changes.
11. Ensure that the Configuration Server parameters in the Recording Processor configuration file are set appropriately. Refer to **Configure the Configuration Server Connection** tab on this page.

> **Important**
>
> The Recording Processor does not support configuration through Genesys Administrator Extension. Configuration is acquired using a local configuration file.

For more information about the RPS options, see Genesys Interaction Recording Options Reference.

## Starting the Recording Processor Script

To launch the RPS, run the following command from the <Recording Processor Install Directory>:

```
<python executable> recording_process.py --config-file=rpconfig.cfg
```

Use the following command line when you want to run multiple instances of RPS on the same machine:

```
<python executable> recording_process.py --config-file=rpconfig.cfg --id=1 --port=8889
```

For each RPS instance, assign a unique id (--id parameter) and port number (--port).

> **Important**
>
> - `--port` defines the server port opened by the RPS process.
> - `--id` represents the suffix of the:
>   - application_name in the configuration file. For example, if application_name is defined in the configuration file as **`RecordingProcessorScript`** and `--id 2` is specified in the command line, then the application object named **`RecordingProcessorScript_2`** will be used to start the program.
>   - log files
>   - metadata json files created in the failed folder
>   - database file created by the process

By default the RPS log file is stored in the working directory. This can be changed by specifying a preexisting folder in the `logfile_path` parameter in the log file section of the configuration file. For example, in Windows:
`logfile_path = C:\logs\recordingProcessor`

# Recording Processor Script (Python 3) RHEL 7

## Important

Voice Processor, a multi-threaded microservice based on the Node.JS platform, is an alternative to the Recording Processor Script (RPS).

- For information on deploying Voice Processor, see Deploying Voice Processor.
- To migrate from an existing RPS deployment to Voice Processor, see Migrating from RPS to Voice Processor.
- RPS is not supported for deployments integrated with SIP Cluster. If your deployment uses SIP Cluster, you must use Voice Processor.

For new deployments, Genesys recommends using Voice Processor instead of RPS.

## Contents

## Prerequisites

Before installing and configuring the RPS, you must have the following prerequisites:

- An Interaction Recording Web Services 8.5.205.32 (or higher) instance where the call recording and screen recording metadata is stored.
- A Recording Crypto Server 8.5.095.16 (or higher) instance to decrypt the encrypted recordings.
- Network access to the WebDAV storage where the recordings are stored.

## Installing Recording Processor Script

### Installing on Windows

1. Install 64 bit Python 3.11.5 from the Python website.
2. Install the **RPS IP** with the installer. **Note:** Install the following third-party libraries in the order they appear and unzip the files in `Administrator` mode.
3. Unzip the <RPS>`\thirdparty\more-itertools-10.1.0.zip` file.
4. Run `py -m pip install .` from the <RPS>`\thirdparty\more-itertools-10.1.0` directory.
5. Unzip the <RPS>`\thirdparty\jaraco.functools-4.0.0.zip` file.
6. Run `py -m pip install .` from the <RPS>`\thirdparty\jaraco.functools-4.0.0` directory.
7. Unzip the <RPS>`\thirdparty\cheroot-10.0.0.zip` file.
8. Run `py -m pip install .` from the <RPS>`\thirdparty\cheroot-10.0.0` directory.
9. Unzip the <RPS>`\thirdparty\web.py-0.62.zip` file.
10. Run `py -m pip install .` from the <RPS>`\thirdparty\web.py-0.62` directory.
11. Unzip the <RPS>`\thirdparty\pyparsing-3.1.1.zip` file.
12. Run `py -m pip install .` from the <RPS>`\thirdparty\pyparsing-3.1.1` directory.
13. Unzip the <RPS>`\thirdparty\httplib2-0.22.0.zip` file.
14. Run `py -m pip install .` from the <RPS>`\thirdparty\httplib2-0.22.0` directory.
15. Unzip the <RPS>`\thirdparty\six-1.16.0.zip` file.
16. Run `py -m pip install .` from the <RPS>`\thirdparty\six-1.16.0` directory.
17. Unzip the <RPS>`\thirdparty\python-dateutil-2.8.2.zip` file.
18. Run `py -m pip install .` from the <RPS>`\thirdparty\python-dateutil-2.8.2` directory.

## Installing on Linux (RHEL)

1. Install `zlib-devel (yum install zlib-devel)`.
2. Install `sqlite devel (yum install sqlite-devel.x86_64)`.
3. Install `libffi devel (yum install libffi-devel)`.
4. Install OpenSSL 1.1.1.
   - For RHEL 7:
     1. Download OpenSSL 1.1.1 from OpenSSL website and compile it. Example config command - `./config --prefix=/usr/home/openssl-1.1.1 --openssldir=/usr/home/openssl-1.1.1`
     2. Add OpenSSL lib path in LD_LIBRARY_PATH. Example command - `export LD_LIBRARY_PATH=/usr/home/openssl-1.1.1/lib:$LD_LIBRARY_PATH`
5. Install 64 bit Python 3.11.5 compiled with OpenSSL 1.1.1 from the Python website.
   - While compiling Cpython 3.11.5 with custom openssl, use `--with-openssl` flag while compilation. Example config command - `./configure --with-openssl=/usr/home/openssl-1.1.1 --enable-optimizations`
6. Install the **RPS IP** with the installer. **Note:** Install the following third-party libraries in the order they appear.
7. Untar the <RPS>`/thirdparty/more-itertools-10.1.0.tar.gz` file.
8. Run `python3 -m pip install .` from the <RPS>`/thirdparty/more-itertools-10.1.0` directory.
9. Untar the <RPS>`/thirdparty/jaraco.functools-4.0.0.tar.gz` file.
10. Run `python3 -m pip install .` from the <RPS>`/thirdparty/jaraco.functools-4.0.0` directory.
11. Untar the <RPS>`/thirdparty/cheroot-10.0.0.tar.gz` file.
12. Run `python3 -m pip install .` from the <RPS>`/thirdparty/cheroot-10.0.0` directory.
13. Untar the <RPS>`/thirdparty/web.py-0.62.tar.gz` file.
14. Run `python3 -m pip install .` from the <RPS>`/thirdparty/web.py-0.62` directory.
15. Untar the <RPS>`/thirdparty/pyparsing-3.1.1.tar.gz` file.
16. Run `python3 -m pip install .` from the <RPS>`/thirdparty/pyparsing-3.1.1` directory.
17. Untar the <RPS>`/thirdparty/httplib2-0.22.0.tar.gz` file.
18. Run `python3 -m pip install .` from the <RPS>`/thirdparty/httplib2-0.22.0` directory.
19. Untar the <RPS>`/thirdparty/six-1.16.0.tar.gz` file.
20. Run `python3 -m pip install .` from the <RPS>`/thirdparty/six-1.16.0` directory.
21. Untar the <RPS>`/thirdparty/python-dateutil-2.8.2.tar.gz` file.
22. Run `python3 -m pip install .` from the <RPS>`/thirdparty/python-dateutil-2.8.2` directory.

> **Important**

- GIR does not support direct upgrade of RPS from Python 2 to Python 3.
- Do not use the `setup.py install` command for installing libraries, instead use `pip install` command as mentioned above.

# Configuring Recording Processor Script

This section describes how to configure the Recording Processor Script for your environment.

## Configuring High Availability

### Recording Processor Cluster

RPS now provides High Availability support using multiple instances of RPS (all active). These active/active instances must be accessed through an HA proxy or load balancer. In this mode, each RPS is responsible for sending metadata to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner based on the load it receives. Each Recording Processor is responsible for fetching metadata from all ICON DB Servers. Media Control Platform (MCP) instances must be configured to access the Recording Processor instances by specifying a single URL that points to the load balancer.

To configure HA:

1. In each Recording Processor's **rpconfig.cfg** configuration file, in the **[processing]** section, set the following options:
   - **get_from_httc_before_posting** = 1
   - **mode** = `active`
2. Ensure that all Recording Processor instances have the *same* network related configuration.

   **Important**

   Genesys recommends that multiple Recording Processor instances be deployed on a single host to optimize the available CPU and take advantage of parallel processing. Multiple Recording Processor instances can then be deployed on other hosts as needed.

3. Using Genesys Administrator Extension, under the **Recording** tab in the IVR Profile, set the **Recording Processor URI** parameter to the load balancer's URL.
4. Configure the load balancer to balance traffic to the Recording Processor instances.

The following is an example configuration section that is needed for setting up an Apache load

balancer for a three-instance Recording Processor cluster.

```
  ProxyPass /cluster balancer://nodecluster
    <Proxy balancer://nodecluster>
        BalancerMember http://<IP address of the active1 Recording Processor server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the active2 Recording Processor Server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the active3 Recording Processor Server>:<active
Recording Processor port>
    </Proxy>
```

> **Important**
>
> SpeechMiner version 8.5.2 or later is required for the Recording Processor cluster support to work properly.

### Recording Processor Script Active/Backup HA

RPS can also provide High Availability support by using two RPS instances (active and backup) accessed through an HA proxy or load balancer in failover mode. In this mode, the active RPS is always responsible for sending metadata to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner, and the backup instance is responsible for receiving and temporarily storing metadata if the active instance is unavailable. Once the active instance recovers, the balancer will direct clients to the active instance, and the backup instance will send any stored data to the active instance for metadata processing. Media Control Platform (MCP) instances must be configured to access the Recording Processor instances by specifying a single URL that points to the load balancer.

To configure HA:

1. In the active Recording Processor's **rpconfig.cfg** configuration file located in the **[processing]** section, set the **mode** parameter to `active`.
2. In the backup Recording Processor's **rpconfig.cfg** configuration file:
   - Set the **mode** parameter to `backup`.
   - In the **[processing]** section, set the **post_uri** parameter to `http://<active_rp_ip>:<active_rp_port>/api/contact-centers/%s/recordings/`.
3. Using Genesys Administrator Extension, under the **Recording** tab in the IVR Profile, set the Recording Processor URI parameter to the failover load balancer's URL.
4. Configure the load balancer to direct traffic to the active Recording Processor instance first and to the backup instance if an error/failure occurs.

The following is an example configuration section that is needed for setting up an Apache load balancer in failover mode for Recording HA support.

```
   ProxyPass /cluster balancer://nodecluster
    <Proxy balancer://nodecluster>
        BalancerMember http://<IP address of the active Recording Processor server>:<active
Recording Processor port>
```

```
        BalancerMember http://<IP address of the backup Recording Processor Server>:<active
Recording Processor port> status=H
    </Proxy>
```

For more information about how to use Genesys Administrator Extension to configure your Contact Center, see the Genesys Administrator Extension Help.

## Configure Passwords

> ### Important
>
> In a Linux or Windows environment, RPS supports reading the environment variables for password related configuration parameters in order to avoid storing the password in plain-text in the configuration file. When both are available, the environment variables take precedence.
>
> The following definitions describe the mapping of the environment variables to the corresponding configuration parameter:
>
> **HTCC_PASSWORD** – maps to the existing configuration parameter under the **htcc** section, `password` value.
> **AUTH_PASSWORD** - maps to the existing configuration parameter under the **auth** section, `password` value.
> **CONFIG_SERVER_PASSWORD** - maps to the existing configuration parameter under the **config_server** section, `password` value.
> **<ICON_ID>_DB_INFO_PASSWORD** - maps to the existing configuration parameter under the **<ICON_ID>_db_info** section, `password` value, where `<ICON_ID>` refers to the ICON instance listed in the **icon_db_servers** section.
>
> For example, if you have `VCCSIPSwitch: icon1` the environment variable that corresponds to the `icon1_db_info` password is `icon1_DB_INFO_PASSWORD`.

In a Windows environment only, the Recording Processor Script (RPS) can store passwords in the Windows Vault instead of in the **rpconfig.cfg** file or requiring the use of environment variables.

For example, run the following command for the Recording Processor Script credentials located at `<Recording Processor Directory>\rp`. This command will prompt the user to enter valid values for the password/key configuration parameters and stores the passwords in the encrypted file named **rp.secret**:

### Command to store:

```
encryptPassword.bat -password <password_string>
```

Where `<password_string>` is a comma-delimited series of key/value pairs, use the format `<environment variable name 1>=<environment variable value 1>,<environment variable name 2>=<environment variable value 2>,<environment variable name 3>=<environment variable value 3>`, and so on. Note that space is not allowed in `<password_string>`.

For example:

```
encryptPassword.bat -password "HTCC_PASSWORD=somepassword1, AUTH_PASSWORD=somepassword2,
CONFIG_SERVER_PASSWORD=somepassword3, ICON1_DB_INFO_PASSWORD=somepassword4,
ICON2_DB_INFO_PASSWORD=somepassword5"
```

> **Important**
> Passwords used with this command cannot contain a comma or an equals sign.

## Configure the Configuration Server Connection

To configure the Configuration Server connection, set the following parameters in the **[config_server]** section of the Recording Processor **rpconfig.cfg** configuration file:

| Parameter Name | Default Value | Description |
|---|---|---|
| application_name | Empty | Specifies the name of the RPS application object in the Configuration Server, when using RPS as a third party server application. |
| hostname | `<ip address>` | Specifies the IP address of the primary Configuration Server. |
| port | `2020` | Specifies the port of the primary Configuration Server. |
| username | `default` | Specifies the Configuration Server username. |
| password | `password` | Specifies the Configuration Server password.<br><br>**Note:**<br>The password can be overridden by the **CONFIG_SERVER_PASSWORD** environment variable. |
| backup_host | Empty | Specifies the IP address of the backup Configuration Server. |
| backup_port | Empty | Specifies the backup port of the backup Configuration Server. |

> **Important**
> Recording Processor Script does not support a secure connection to the Configuration Server.

## Configuring the Server Port

In the **[rp_server]** section of the **rpconfig.cfg** file, set the **port** parameter.

> **Important**
>
> You can also set the "port" parameter using the command line with the --port command line argument. The command line argument takes precedence over the configuration file value.

## Configuring the Connection to Interaction Recording Web Services (Web Services)

To configure the Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) connection, set the following parameters in the **[htcc]** section of the Recording Processor **rpconfig.cfg** configuration file:

| Parameter Name | Default Value | Description |
|---|---|---|
| base_uri | `http://<Web Services IP>:<Web Services Port>` | Specifies the Base URI for accessing the Interaction Recording Web Services (Web Services) API. |
| username | `ops` | Specifies the username used to access the Interaction Recording Web Services (Web Services) account. |
| password | `ops` | Specifies the password used to access the Interaction Recording Web Services (Web Services) account.<br><br>**Note:** The password can be overridden by the **HTCC_PASSWORD** environment variable. |

Each Interaction Recording Web Services (Web Services) instance must have a region associated with it. Set the region parameter in the [metadata] section of the rpconfig.cfg file to match the region associated with Interaction Recording Web Services (Web Services) instance set to receive the Recording Processor's metadata.

## Configuring Cross-Site Request Forgery (CSRF) Protection

If Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) has CSRF enabled, set the following parameter in the **[htcc]** section of the **rpconfig.cfg** file:

- **csrfp** = 1

## Configuring the Connection to SpeechMiner

To configure the SpeechMiner Connection:

1. In the IVR Profile, set the recording destinations to point to the SpeechMiner interaction receiver:
   a. Login to Genesys Administrator Extension, and navigate to **Configuration > System >**

**Configuration Manager**.

b. Under **Voice Platform**, select **Voice Platform Profiles**.

c. Click on the IVR Profile for which you want to set the recording destination.

d. Select the **Recording** tab.

e. In the**SpeechMiner Interaction Receiver** field, enter the URL that points to the SpeechMiner service responsible for accepting metadata from the RPS for this profile. For example, `https://<SpeechMiner IP>/interactionreceiver`.

f. In the **SpeechMiner Interaction Receiver Authorization Header** field, enter the authorization information (username:password) required to connect to the SpeechMiner service used by the RPS. For example, `user:password`.

> **Important**
>
> The values of these options must match the corresponding configuration options in the SpeechMiner system.

## Configuring Failed Message Files

The Recording processor can backup messages that fail to POST correctly to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner. These files are located in the **<recording processor dir>\failed** folder.

In the **rpconfig.cfg** configuration file, add the following parameter:

```
[processing]
backup_failed_metadata = 1
```

## Configuring the Agent Hierarchies

Recording Processor Script uses the agent hierarchy information to set the access control information for recordings within the recording metadata. Refer to Access Control for Genesys Interaction Recording Users to configure this appropriately.

## Configuring Basic Authorization

In the **rpconfig.cfg** configuration file, set the following parameters:

```
[auth]
# Basic Authentication username and password. Set username blank to disable.
username = rp_username
password = rp_password
```

> **Important**

- The username and password must match the username and password entered in the IVR Profile. For more information about configuring the IVR Profile, see the IVR Profile section.
- The password can be overridden by the **AUTH_PASSWORD** environment variable.

## Configuring After Call Work

Recording Processor can collect After Call Work (ACW) customized data from ICON.

In the **rpconfig.cfg** file, in the **[processing]** section, add the following parameters:

- **enable_acw**—Set it to 1.
- **acw_threshold_minutes**—Set it to the maximum time to wait for the customized attached data.

### Important

- If Call Customized Attached Data is still not available in the ICON database after *acw_threshold_minutes*, the RPS will stop collecting customized data for this recording and write it to the database.
- If *enable_acw* is set to `0`, ACW customized data will not be included.
- If disposition code is required in the metadata, you must set **enable_acw** and **acw_threshold_minutes** using Recording Processor configuration.
  The disposition code is part of the user data collected during ACW. For this reason, **enable_acw** must be enabled in the Recording Processor. If it is not enabled, the data will not be collected.
  If the disposition code must be collected from the Recording Processor, configure the following to include the disposition code for recording:
  `[processing] enable_acw=1`
  `[metadata] acw_threshold_minutes=5`. Where 5 is the maximum time (in minutes) to wait for the disposition code.
  In the ICON configuration, the **EventData** parameter in the **custom-states** section, must include `char,DispositionCode` and **store-event-data** must be set to `conf` to collect the attached data:
  `[custom-states] store-event-data=conf EventData=char,DispositionCode`
  For additional information, refer to the ICON Deployment Guide.

## Configuring ICON for Recording Processor

### Important

When configuring Recording Processor to connect to a primary and backup ICON Database in HA mode, two separate DB Servers must be used. The DB Servers must run in an active/active pair mode.

To configure ICON, edit the **rpconfig.cfg** configuration file as follows:

1. Configure the switches:

   Add a configuration option for each switch name under the **[icon_db_servers]** section. You can specify more than two ICON databases per SIP Switch configuration. For example:

   ```
   [icon_db_servers]
   SIP_Switch1: icon1
   SIP_Switch2: icon2, icon2Backup
   SIP_Switch3: icon3, icon4, icon5, icon6
   ```

   **Important**

   In the above example, **SIP_Switch3** has 4 ICON databases. The Recording Processor Script (RPS) keeps track of the ICON database instance currently used. If the current database instance becomes unavailable, RPS will attempt the operation in the next database.

   The configuration option name must match the exact name of the switch as configured in the Genesys configuration. The primary and backup ICON names must be unique, but do not have to match anything in the Genesys configuration.

2. Configure the ICON Connection Settings:

   - For each unique ICON specified in the first step, create a new section using the following syntax: *<ICON_ID>*_db_info, where *<ICON_ID>* corresponds to the values defined in the **[icon_db_servers]** section above.
   - **dbengine** must be `mssql`, `oracle`, `db2`, or `postgres`.
   - **dbserver_host** and **dbserver_port** specify the host and port information for the Genesys DB Server.
   - **dbms** specifies the host where the database resides.

   The following is an example using the values for **SIP_Switch1** and **SIP_Switch2** from step 1:

   ```
   [icon1_db_info]
   dbserver_host = vm221.us.int.genesyslab.com
   dbserver_port = 12201
   username = iconuser_1
   password = genesys
   dbname   = ICON_LRM_DB_1
   dbms = 10.0.0.228,1433
   dbengine = mssql
   ```

```
[icon2_db_info]
dbserver_host = vm222.us.int.genesyslab.com
dbserver_port = 12201
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon2Backup_db_info]
dbserver_host = vm223.us.int.genesyslab.com
dbserver_port = 12201
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon_oracle_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname =
dbms = <database host/Oracle SID>
dbengine = oracle

[icon_postgres_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname = <database name>
dbms = <database host>
dbengine = postgres

[icon_db2_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname =
dbms = <database host>
dbengine = db2
```

### Important

- For Oracle or DB2 implementations, the **dbname** parameter must be left blank or empty.
- The password can be overridden by the **<ICON_ID>_DB_INFO_PASSWORD** environment variable.

In the example above, the RPS will use the connection properties in section **[icon1_db_info]** when processing recording metadata from an MCP provisioned to SIP_Switch1. The RPS will use the connection properties in section **[icon2_db_info]** when processing recording metadata from an MCP provisioned to SIP_Switch2. In the case of SIP_Switch2, the RPS will use the connection settings in

**[icon2Backup_db_info]** if the primary ICON (icon2) is unavailable when recording metadata is being processed.

## Configure how to Filter Metadata from ICON

The Recording Processor supports the ability to filter specific attached data fields (based on the key name), such as attached data and After Call Work (ACW) customized data retrieved from the ICON database. This support prevents specific metadata from reaching additional GIR related components (for example, SpeechMiner).

The following two sections describe how to:

- Filter attached data.
- Filter ACW.

### Important

- Verify that the following items are not removed from the filter. Removing these items may cause errors in GIR:
  - GRECORD_PARTITIONS
  - GRECORD_PROGRAM
  - GSIP_REC_FN
- When running SpeechMiner, you must include Workspace Web Edition (WWE) in the attached `attached_data_filter` and `acw_custom_data_filter` Recording Processor configuration values. For example:

```
[filter]
attached_data_filter=^ORSI:|^WWE
acw_custom_data_filter=^ORSI:|^WWE
```

### Filter Attached Data

1. Edit the **rpconfig.cfg** file.
2. Locate the **Filter** section. If the **Filter** section does not exist, add it as follows:

```
...
[filter]
...
```

3. Add a new option called **attached_data_filter** to the **Filter** section as follows. The value must be a Regex pattern used to filter out attached data whose key name matches the pattern.

```
...
[filter]
attached_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
```

```
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/3.11/library/re.html.

4. Add a new option called **attached_data_filter_exception** to this section as follows. The value must be a Regex pattern used to exclude key names that should not be filtered out (for example, like GRECORD_PARTITIONS).

```
...
[filter]
attached_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
attached_data_filter_exception = ^GRECORD_PARTITIONS$     ;   (Note: this is the default
value when the option is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/3.11/library/re.html.

5. Restart the Recording Processor.

## Filter ACW Related Custom Data

1. Edit the **rpconfig.cfg** file.
2. Locate the **Filter** section. If the **Filter** section does not exist, add it as follows:

```
...
[filter]
...
```

3. Add a new option called **acw_custom_data_filter** to the **Filter** section as follows. The value must be a Regex pattern used to filter out ACW whose key name matches the pattern.

```
...
[filter]
acw_custom_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/3.11/library/re.html.

4. Add a new option called **acw_custom_data_filter_exception** to this section as follows. The value must be a Regex pattern used to exclude ACW that should not be filtered out (for example, like GRECORD_PARTITIONS).

```
...
[filter]
acw_custom_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
acw_custom_data_filter_exception = ^GRECORD_PARTITIONS$     ;   (Note: this is the
default value when the option is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/3.11/library/re.html.

5. Restart the Recording Processor.

## Configuring SSL for Recording Processor

To configure SSL:

### Configure HTTPS on the Primary Recording Processor Server

1. Create a self-signed certificate and private key for the Recording Processor host. For example, on RHEL run: `openssl req -new -x509 -days 1024 -nodes -out cert228.pem -keyout cert228key.pem`
2. In the `rp_server` section of the Recording Processor's configuration file, set the following parameters:
   - `ssl_certificate`—To point to the certificate PEM file. For example, `ssl_certificate=cert228.pem`.
   - `ssl_private_key`—To point to the private key file. For example, `cert228.pem`.
3. Give the self-signed certificate PEM file to any MCP client that needs to validate the certificate during the SSL handshake. See the "Enable Secure Communication" section Genesys Voice Platform 8.5 User's Guide.
4. Restart Recording Processor.

### Configure the HTTPS connection to Interaction Recording Web Services (Web Services)

1. Set up HTTPS on Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier). See the Genesys Security Deployment Guide.
2. Get the corresponding certificate for the Interaction Recording Web Services (Web Services) server. Set the caCertificate option in your Interaction Recording Web Services application (see caCertificate if you're using a Web Services application).
3. In the **[htcc]** section of the Recording Processor configuration file, set **base_uri** parameter to use `https`.
4. In the **[client]** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA certificate file.

> **Important**
>
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

### Configure the HTTPS connection to SpeechMiner

1. Set up HTTPS on SpeechMiner.See the Genesys Security Deployment Guide.
2. Set the **disable_ssl_certificate_validation** parameter in the **[speechminer]** section of the Recording Processor configuration to a value of 1.
3. Using Genesys Administrator Extension on the `Recording` tab of the IVR Profile, modify the `SpeechMiner Interaction Receiver` field use `https` as the protocol in the URL.
4. In the **[client]** section, set the **certs** parameter to point to the file that contains the certificate (see

previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA cert file.

> **Important**
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

### Configure the HTTPS connection from the backup Recording Processor to the primary Recording Processor

1. Configure HTTPS on the primary Recording Processor.
2. Get the corresponding PEM certificate for the Web Services server.
3. In the **[processing]** section of the Recording Processor configuration file, set the **post_uri** parameter to use `https` as the protocol in the URL.
4. In the **client** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA certificate file.

> **Important**
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

## Configure the HTTPS on the backup Recording Processor Server

Follow the same procedure used for the Primary Recording Processor Server using a new certificate and private key for the Backup Recording Processor's server.

## Configuring the IVR Profile

Using Genesys Administrator Extension, configure the following parameters on the **Recording** tab of the IVR Profile:

1. **Recording Processor URI**—The URI that the Media Control Platform (MCP) uses to post the metadata of the audio recording after the recording is complete. For example, `http:// <Recording Processor Host>/api/contact-centers/<Contact Cente Domain Name>/recordings/`.

> **Important**
> The value for the URI must always end with a forward slash (/).

2. SpeechMiner Interaction Receiver—The URL that points to the SpeechMiner service responsible for accepting metadata from the RPS for this profile.
3. SpeechMiner Interaction Receiver Authorization Header—The authorization information required to connect to the SpeechMiner service used by the RPS. For example, `<SpeechMiner Webserver Username>:<SpeechMiner Webserver Password>`.

For more information, see the Configuring GVP.

## Configuring the Recording Processor Using Genesys Administrator Extension (Optional)

The Recording Processor uses a configuration file instead of a specific application object in Configuration Server. However, it is possible to configure the Recording Processor as a "third-party server" application enabling Genesys Administrator Extension to monitor, start, and stop the process.

The following steps describe how to setup Recording Processor as a "third party server" application in Genesys Administrator Extension. For more information, see the "Using the Management Layer" section of the Framework 8.5.1 Management Layer User's Guide

Configuring RPS to Start/Stop via LCA using Genesys Administrator Extension:

1. Install and deploy the latest RPS.
2. Make sure that the Local Control Agent (LCA) is running.
3. Create a new application template in Genesys Administrator Extension called `Recording Processor Script` of type `Third Party Server`.
4. Create a new application (for example, `myRPS`) in Genesys Administrator Extension using this new application template.
5. Set the `Command Line` parameter (for example, C:\Python311\python.exe).
6. Set the `Host` parameter in the application's server info to the correct Host object.
7. Set the `Working Directory` parameter to the `<Recording Processor Install Directory>\rp` directory. For example, `/opt/genesys/Recording_Processor_Script_8.5/rp/`.
8. Set the `Command Line Arguments` parameter to the appropriate values. For example, `recording_process.py --config-file=/opt/genesys/Recording_Processor_Script_8.5/rp/rpconfig.cfg`.
   Refer to the Starting the Recording Processor Script section for additional command line parameters
9. Make sure that LCA has permission to read and write to the Recording Processor installation directory and Recording Processor log directory.
10. Save the configuration changes.
11. Ensure that the Configuration Server parameters in the Recording Processor configuration file are set appropriately. Refer to **Configure the Configuration Server Connection** tab on this page.

> **Important**
>
> The Recording Processor does not support configuration through Genesys

> Administrator Extension. Configuration is acquired using a local configuration file.

For more information about the RPS options, see Genesys Interaction Recording Options Reference.

## Starting the Recording Processor Script

To launch the RPS, run the following command from the <Recording Processor Install Directory>:

```
<python executable> recording_process.py --config-file=rpconfig.cfg
```

Use the following command line when you want to run multiple instances of RPS on the same machine:

```
<python executable> recording_process.py --config-file=rpconfig.cfg --id=1 --port=8889
```

For each RPS instance, assign a unique id (--id parameter) and port number (--port).

> ### Important
>
> - `--port` defines the server port opened by the RPS process.
> - `--id` represents the suffix of the:
>   - application_name in the configuration file. For example, if application_name is defined in the configuration file as **`RecordingProcessorScript`** and `--id 2` is specified in the command line, then the application object named **`RecordingProcessorScript_2`** will be used to start the program.
>   - log files
>   - metadata json files created in the failed folder
>   - database file created by the process

By default the RPS log file is stored in the working directory. This can be changed by specifying a preexisting folder in the `logfile_path` parameter in the log file section of the configuration file. For example, in Windows:
`logfile_path = C:\logs\recordingProcessor`

## Recording Processor Script Legacy (Python 2) Deprecated

### Important

Recording Processor Script Legacy (based on Python 2) has been discontinued as of March 31, 2024.

### Important

Voice Processor, a multi-threaded microservice based on the Node.JS platform, is an alternative to the Recording Processor Script (RPS).

- For information on deploying Voice Processor, see Deploying Voice Processor.
- To migrate from an existing RPS deployment to Voice Processor, see Migrating from RPS to Voice Processor.
- RPS is not supported for deployments integrated with SIP Cluster. If your deployment uses SIP Cluster, you must use Voice Processor.

For new deployments, Genesys recommends using Voice Processor instead of RPS.

## Contents

Genesys Interaction Recording (GIR) needs the Recording Processor Script (RPS) to manage the recording metadata between Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner.

## Prerequisites

Before installing and configuring the RPS, you must have the following prerequisites:

- An Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) instance where the call recording and screen recording metadata is stored.
- A Recording Crypto Server instance to decrypt the encrypted recordings.
- Network access to the WebDAV storage where the recordings are stored.

## Installing Recording Processor Script

### Installing on Windows

1. Install 32 bit Python 2.7.5 or latest 2.7.x release from the Python website.
2. Install the RPS IP.
3. Unzip the <RPS>\thirdparty\httplib2-0.8.zip file.
4. From the newly created directory, run: `python setup.py install`.
5. Unzip the <RPS>\thirdparty\setuptools-1.3.2.zip file.
6. From the newly created directory, run: `python setup.py install`.
7. Unzip the <RPS>\thirdparty\python-dateutil-1.5.zip file.
8. From the newly created directory, run: `python setup.py install`.
9. Unzip the <RPS>\thirdparty\web.py-0.37.zip file.
10. From the newly created directory, run: `python setup.py install`.
11. Download pyOpenSSL for Windows (32 bit) from the Python pyOpenSSL site.
12. Install pyOpenSSL by running `pyOpenSSL-0.12.1.win32-py2.7.exe`.

> **Important**
>
> Installing pyOpenSSL (the previous steps) is optional, and is only required if an HTTPS server is needed for Recording Processor Script to receive metadata. Without it, only an HTTP server is supported.

## Installing on Linux (RHEL)

1. Install `zlib-devel (yum install zlib-devel)`.
2. Install `sqlite devel (yum install sqlite-devel.x86_64)`.
3. Install `openssl devel (yum install openssl-devel.x86_64)`.
4. Install Python 2.7.5 or latest 2.7.x release from the Python website:
   - Genesys recommends that newer versions of Python are installed separately from existing versions (do not update).
   - See the Example below for an example of how to install CPython 2.7.6 on RHEL5.
5. Install/deploy the RPS IP.
6. Install `httplib2-0.8`:
   a. Untar `httplib2-0.8.tar.gz` from the thirdparty directory in the RPS installation directory.
   b. From the newly created directory, run: `python setup.py install`.

3. Install `setuptools-1.3.2`:
   a. Untar `setuptools-1.3.2.tar.gz` from the thirdparty directory in the RPS installation directory.
   b. From the newly created directory, run: `python setup.py install`.

3. Install `python-dateutil-1.5`:
   a. Untar `python-dateutil-1.5.tar.gz` from the thirdparty directory in the RPS installation directory.
   b. From the newly created directory, run: `python setup.py install`.

3. Install `web.py-0.37`:
   a. Untar `web.py-0.37.tar.gz` from the thirdparty directory in the RPS installation directory.
   b. From the newly created directory, run: `python setup.py install`.

3. Install `pyOpenSSL-0.12`:
   a. Download `pyOpenSSL-0.12.tar.gz` from the pyOpenSSL 0.12 site.
   b. Untar the downloaded file, `pyOpenSSL-0.12.tar.gz` to the RPS installation directory.
   c. From the newly created directory, run the following command to build the library: `python setup.py build`.
   d. Install the library, run: `python setup.py install`.

> **Important**
> Installing pyOpenSSL (the previous steps) is optional, and is only required if an HTTPS server is needed for Recording Processor Script to receive metadata. Without it, only an HTTP server is supported.

> RPS on RHEL8 cannot build pyOpenSSL-0.12. You must download **openssl-1.0.1e.tar.gz** from https://ftp.openssl.org/source/old/1.0.1 , build openssl-1.0.1e, proceed to build Python 2.7.18 normally, and then build pyOpenSSL-0.12 while also updating LD_LIBRARY_PATH to point to the openssl-1.0.1e libraries.

### Example: Installing CPython 2.7.6 on RHEL5 (64bit)

The following instructions are intended as an example only. A specific system or environment may require different steps when installing CPython 2.7.6 on RHEL5 (64bit):

1. Verify that `zlib-devel` is installed on the OS (yum install `zlib-devel`).
2. Verify that `sqlite dev` is installed on the OS (yum install `sqlite-devel.x86_64`).
3. Verify that `openssl devel` is installed on the OS (yum install `openssl-devel.x86_64`).
4. Download CPython 2.7.6 source from the Python site.
5. Untar compressed source.
6. Run "`./configure --enable-ipv6`".
7. Run "'make altinstall'" (this should prevent the overwriting of any existing versions).

## Upgrading Recording Processor Script

1. Stop the RPS process.
2. Stop the RPS application.
3. Back up the RPS configuration file (`rpconfig.cfg`) and the sqlite file from the `\rp` directory (`rpqueue.db`).
4. Rename the existing installation folder name to `<folder name>.<old.current_date>` or something similar.
5. Uninstall the RPS component.
6. Install the new RPS component.
7. Copy the `rpconfig.cfg` and `rpqueue.db` files from the previous version into the `\rp` folder inside the new installation directory.
8. Start the new RPS application.
9. Repeat the above steps for additional RPS instances.

# Configuring Recording Processor Script

This section describes how to configure the Recording Processor Script for your environment.

## Configuring High Availability

### Recording Processor Cluster

RPS now provides High Availability support using multiple instances of RPS (all active). These active/active instances must be accessed through an HA proxy or load balancer. In this mode, each RPS is responsible for sending metadata to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner based on the load it receives. Each Recording Processor is responsible for fetching metadata from all ICON DB Servers. Media Control Platform (MCP) instances must be configured to access the Recording Processor instances by specifying a single URL that points to the load balancer.

To configure HA:

1. In each Recording Processor's **rpconfig.cfg** configuration file, in the **[processing]** section, set the following options:
   - **get_from_httc_before_posting** = 1
   - **mode** = `active`
2. Ensure that all Recording Processor instances have the *same* network related configuration.

> **Important**
>
> Genesys recommends that multiple Recording Processor instances be deployed on a single host to optimize the available CPU and take advantage of parallel processing. Multiple Recording Processor instances can then be deployed on other hosts as needed.

3. Using Genesys Administrator Extension, under the **Recording** tab in the IVR Profile, set the **Recording Processor URI** parameter to the load balancer's URL.
4. Configure the load balancer to balance traffic to the Recording Processor instances.

The following is an example configuration section that is needed for setting up an Apache load balancer for a three-instance Recording Processor cluster.

```
  ProxyPass /cluster balancer://nodecluster
    <Proxy balancer://nodecluster>
        BalancerMember http://<IP address of the active1 Recording Processor server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the active2 Recording Processor Server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the active3 Recording Processor Server>:<active
Recording Processor port>
    </Proxy>
```

> **Important**
>
> SpeechMiner version 8.5.2 or later is required for the Recording Processor cluster support to work properly.

### Recording Processor Script Active/Backup HA

RPS can also provide High Availability support by using two RPS instances (active and backup) accessed through an HA proxy or load balancer in failover mode. In this mode, the active RPS is always responsible for sending metadata to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner, and the backup instance is responsible for receiving and temporarily storing metadata if the active instance is unavailable. Once the active instance recovers, the balancer will direct clients to the active instance, and the backup instance will send any stored data to the active instance for metadata processing. Media Control Platform (MCP) instances must be configured to access the Recording Processor instances by specifying a single URL that points to the load balancer.

To configure HA:

1. In the active Recording Processor's **rpconfig.cfg** configuration file located in the **[processing]** section, set the **mode** parameter to `active`.
2. In the backup Recording Processor's **rpconfig.cfg** configuration file:
   - Set the **mode** parameter to `backup`.
   - In the **[processing]** section, set the **post_uri** parameter to `http://<active_rp_ip>:<active_rp_port>/api/contact-centers/%s/recordings/`.
3. Using Genesys Administrator Extension, under the **Recording** tab in the IVR Profile, set the Recording Processor URI parameter to the failover load balancer's URL.
4. Configure the load balancer to direct traffic to the active Recording Processor instance first and to the backup instance if an error/failure occurs.

The following is an example configuration section that is needed for setting up an Apache load balancer in failover mode for Recording HA support.

```
   ProxyPass /cluster balancer://nodecluster
    <Proxy balancer://nodecluster>
        BalancerMember http://<IP address of the active Recording Processor server>:<active
Recording Processor port>
        BalancerMember http://<IP address of the backup Recording Processor Server>:<active
Recording Processor port> status=H
    </Proxy>
```

For more information about how to use Genesys Administrator Extension to configure your Contact Center, see the Genesys Administrator Extension Help.

## Configure Passwords

### Important

In a Linux or Windows environment, RPS supports reading the environment variables for password related configuration parameters in order to avoid storing the password in plain-text in the configuration file. When both are available, the environment variables take precedence.

The following definitions describe the mapping of the environment variables to the corresponding configuration parameter:

**HTCC_PASSWORD** – maps to the existing configuration parameter under the **htcc** section, `password` value.
**AUTH_PASSWORD** - maps to the existing configuration parameter under the **auth** section, `password` value.
**CONFIG_SERVER_PASSWORD** - maps to the existing configuration parameter under the **config_server** section, `password` value.
**<ICON_ID>_DB_INFO_PASSWORD** - maps to the existing configuration parameter under the **<ICON_ID>_db_info** section, `password` value, where <ICON_ID> refers to the ICON instance listed in the **icon_db_servers** section.

For example, if you have `VCCSIPSwitch: icon1` the environment variable that corresponds to the `icon1_db_info` password is `icon1_DB_INFO_PASSWORD`.

In a Windows environment only, the Recording Processor Script (RPS) can store passwords in the Windows Vault instead of in the **rpconfig.cfg** file or requiring the use of environment variables.

For example, run the following command for the Recording Processor Script credentials located at `<Recording Processor Directory>\rp`. This command will prompt the user to enter valid values for the password/key configuration parameters and stores the passwords in the encrypted file named **rp.secret**:

### Command to store:

```
encryptPassword.bat -password <password_string>
```

Where `<password_string>` is a comma-delimited series of key/value pairs, use the format `<environment variable name 1>=<environment variable value 1>,<environment variable name 2>=<environment variable value 2>,<environment variable name 3>=<environment variable value 3>`, and so on. Note that space is not allowed in `<password_string>`.

For example:

```
encryptPassword.bat -password "HTCC_PASSWORD=somepassword1, AUTH_PASSWORD=somepassword2,
CONFIG_SERVER_PASSWORD=somepassword3, ICON1_DB_INFO_PASSWORD=somepassword4,
ICON2_DB_INFO_PASSWORD=somepassword5"
```

### Important

Passwords used with this command cannot contain a comma or an equals sign.

## Configure the Configuration Server Connection

To configure the Configuration Server connection, set the following parameters in the **[config_server]** section of the Recording Processor **rpconfig.cfg** configuration file:

| Parameter Name | Default Value | Description |
|---|---|---|
| application_name | Empty | Specifies the name of the RPS application object in the Configuration Server, when using RPS as a third party server application. |
| hostname | `<ip address>` | Specifies the IP address of the primary Configuration Server. |
| port | `2020` | Specifies the port of the primary Configuration Server. |
| username | `default` | Specifies the Configuration Server username. |
| password | `password` | Specifies the Configuration Server password.<br>**Note:**<br>The password can be overridden by the **CONFIG_SERVER_PASSWORD** environment variable. |
| backup_host | Empty | Specifies the IP address of the backup Configuration Server. |
| backup_port | Empty | Specifies the backup port of the backup Configuration Server. |

> **Important**
>
> Recording Processor Script does not support a secure connection to the Configuration Server.

## Configuring the Server Port

In the **[rp_server]** section of the **rpconfig.cfg** file, set the **port** parameter.

> **Important**
>
> You can also set the "port" parameter using the command line with the --port command line argument. The command line argument takes precedence over the configuration file value.

## Configuring the Connection to Interaction Recording Web Services (Web Services)

To configure the Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) connection, set the following parameters in the **[htcc]** section of the Recording Processor **rpconfig.cfg** configuration file:

| Parameter Name | Default Value | Description |
|---|---|---|
| base_uri | `http://<Web Services IP>:<Web Services Port>` | Specifies the Base URI for accessing the Interaction Recording Web Services (Web Services) API. |
| username | ops | Specifies the username used to access the Interaction Recording Web Services (Web Services) account. |
| password | ops | Specifies the password used to access the Interaction Recording Web Services (Web Services) account.<br><br>**Note:** The password can be overridden by the **HTCC_PASSWORD** environment variable. |

Each Interaction Recording Web Services (Web Services) instance must have a region associated with it. Set the region parameter in the [metadata] section of the rpconfig.cfg file to match the region associated with Interaction Recording Web Services (Web Services) instance set to receive the Recording Processor's metadata.

## Configuring Cross-Site Request Forgery (CSRF) Protection

If Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) has CSRF enabled, set the following parameter in the **[htcc]** section of the **rpconfig.cfg** file:

- **csrfp** = 1

## Configuring the Connection to SpeechMiner

To configure the SpeechMiner Connection:

1. In the IVR Profile, set the recording destinations to point to the SpeechMiner interaction receiver:
   a. Login to Genesys Administrator Extension, and navigate to **Configuration > System > Configuration Manager**.
   b. Under **Voice Platform**, select **Voice Platform Profiles**.
   c. Click on the IVR Profile for which you want to set the recording destination.
   d. Select the **Recording** tab.
   e. In the**SpeechMiner Interaction Receiver** field, enter the URL that points to the SpeechMiner service responsible for accepting metadata from the RPS for this profile. For example, `https://<SpeechMiner IP>/interactionreceiver`.

f. In the **SpeechMiner Interaction Receiver Authorization Header** field, enter the authorization information (username:password) required to connect to the SpeechMiner service used by the RPS. For example, `user:password`.

> **Important**
>
> The values of these options must match the corresponding configuration options in the SpeechMiner system.

## Configuring Failed Message Files

The Recording processor can backup messages that fail to POST correctly to Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier) and SpeechMiner. These files are located in the **<recording processor dir>\failed** folder.

In the **rpconfig.cfg** configuration file, add the following parameter:

```
[processing]
backup_failed_metadata = 1
```

## Configuring the Agent Hierarchies

Recording Processor Script uses the agent hierarchy information to set the access control information for recordings within the recording metadata. Refer to Access Control for Genesys Interaction Recording Users to configure this appropriately.

## Configuring Basic Authorization

In the **rpconfig.cfg** configuration file, set the following parameters:

```
[auth]
# Basic Authentication username and password. Set username blank to disable.
username = rp_username
password = rp_password
```

> **Important**
>
> - The username and password must match the username and password entered in the IVR Profile. For more information about configuring the IVR Profile, see the IVR Profile section.
> - The password can be overridden by the **AUTH_PASSWORD** environment variable.

## Configuring After Call Work

Recording Processor can collect After Call Work (ACW) customized data from ICON.

In the **rpconfig.cfg** file, in the **[processing]** section, add the following parameters:

- **enable_acw**—Set it to 1.
- **acw_threshold_minutes**—Set it to the maximum time to wait for the customized attached data.

### Important

- If Call Customized Attached Data is still not available in the ICON database after *acw_threshold_minutes*, the RPS will stop collecting customized data for this recording and write it to the database.
- If *enable_acw* is set to `0`, ACW customized data will not be included.
- If disposition code is required in the metadata, you must set **enable_acw** and **acw_threshold_minutes** using Recording Processor configuration.
  The disposition code is part of the user data collected during ACW. For this reason, **enable_acw** must be enabled in the Recording Processor. If it is not enabled, the data will not be collected.
  If the disposition code must be collected from the Recording Processor, configure the following to include the disposition code for recording:
  `[processing] enable_acw=1`
  `[metadata] acw_threshold_minutes=5`. Where 5 is the maximum time (in minutes) to wait for the disposition code.
  In the ICON configuration, the **EventData** parameter in the **custom-states** section, must include `char,DispositionCode` and **store-event-data** must be set to `conf` to collect the attached data:
  `[custom-states] store-event-data=conf EventData=char,DispositionCode`
  For additional information, refer to the ICON Deployment Guide.

## Configuring ICON for Recording Processor

### Important

When configuring Recording Processor to connect to a primary and backup ICON Database in HA mode, two separate DB Servers must be used. The DB Servers must run in an active/active pair mode.

To configure ICON, edit the **rpconfig.cfg** configuration file as follows:

1. Configure the switches:

   Add a configuration option for each switch name under the **[icon_db_servers]** section. You can specify more than two ICON databases per SIP Switch configuration. For example:

```
[icon_db_servers]
SIP_Switch1: icon1
SIP_Switch2: icon2, icon2Backup
SIP_Switch3: icon3, icon4, icon5, icon6
```

> **Important**
>
> In the above example, **SIP_Switch3** has 4 ICON databases. The Recording Processor Script (RPS) keeps track of the ICON database instance currently used. If the current database instance becomes unavailable, RPS will attempt the operation in the next database.

The configuration option name must match the exact name of the switch as configured in the Genesys configuration. The primary and backup ICON names must be unique, but do not have to match anything in the Genesys configuration.

2. Configure the ICON Connection Settings:
   - For each unique ICON specified in the first step, create a new section using the following syntax: <*ICON_ID*>_db_info, where <*ICON_ID*> corresponds to the values defined in the **[icon_db_servers]** section above.
   - **dbengine** must be `mssql`, `oracle`, `db2`, or `postgres`.
   - **dbserver_host** and **dbserver_port** specify the host and port information for the Genesys DB Server.
   - **dbms** specifies the host where the database resides.

   The following is an example using the values for **SIP_Switch1** and **SIP_Switch2** from step 1:

```
[icon1_db_info]
dbserver_host = 10.0.0.221
dbserver_port = 12201
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon2_db_info]
dbserver_host = 10.0.0.222
dbserver_port = 12201
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon2Backup_db_info]
dbserver_host = 10.0.0.223
dbserver_port = 12201
username = iconuser_1
password = genesys
dbname   = ICON_LRM_DB_1
dbms = 10.0.0.228,1433
dbengine = mssql

[icon_oracle_db_info]
dbserver_host = <host>
dbserver_port = <port>
```

```
username = <username>
password = <password>
dbname =
dbms = <database host/Oracle SID>
dbengine = oracle

[icon_postgres_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname = <database name>
dbms = <database host>
dbengine = postgres

[icon_db2_db_info]
dbserver_host = <host>
dbserver_port = <port>
username = <username>
password = <password>
dbname =
dbms = <database host>
dbengine = db2
```

### Important

- For Oracle or DB2 implementations, the **dbname** parameter must be left blank or empty.
- The password can be overridden by the **<ICON_ID>_DB_INFO_PASSWORD** environment variable.

In the example above, the RPS will use the connection properties in section **[icon1_db_info]** when processing recording metadata from an MCP provisioned to `SIP_Switch1`. The RPS will use the connection properties in section **[icon2_db_info]** when processing recording metadata from an MCP provisioned to `SIP_Switch2`. In the case of `SIP_Switch2`, the RPS will use the connection settings in **[icon2Backup_db_info]** if the primary ICON (icon2) is unavailable when recording metadata is being processed.

## Configure how to Filter Metadata from ICON

The Recording Processor supports the ability to filter specific attached data fields (based on the key name), such as attached data and After Call Work (ACW) customized data retrieved from the ICON database. This support prevents specific metadata from reaching additional GIR related components (for example, SpeechMiner).

The following two sections describe how to:

- Filter attached data.
- Filter ACW.

### Important

- Verify that the following items are not removed from the filter. Removing these items may cause errors in GIR:
  - GRECORD_PARTITIONS
  - GRECORD_PROGRAM
  - GSIP_REC_FN
- When running SpeechMiner, you must include Workspace Web Edition (WWE) in the attached `attached_data_filter` and `acw_custom_data_filter` Recording Processor configuration values. For example:

```
[filter]
attached_data_filter=^ORSI:|^WWE
acw_custom_data_filter=^ORSI:|^WWE
```

## Filter Attached Data

1. Edit the **rpconfig.cfg** file.
2. Locate the **Filter** section. If the **Filter** section does not exist, add it as follows:

```
...
[filter]
...
```

3. Add a new option called **attached_data_filter** to the **Filter** section as follows. The value must be a Regex pattern used to filter out attached data whose key name matches the pattern.

```
...
[filter]
attached_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
...
```

   For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/2/library/re.html.

4. Add a new option called **attached_data_filter_exception** to this section as follows. The value must be a Regex pattern used to exclude key names that should not be filtered out (for example, like GRECORD_PARTITIONS).

```
...
[filter]
attached_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
attached_data_filter_exception = ^GRECORD_PARTITIONS$     ;   (Note: this is the default
value when the option is not specified.)
...
```

   For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/2/library/re.html.

5. Restart the Recording Processor.

### Filter ACW Related Custom Data

1. Edit the **rpconfig.cfg** file.
2. Locate the **Filter** section. If the **Filter** section does not exist, add it as follows:

```
...
[filter]
...
```

3. Add a new option called **acw_custom_data_filter** to the **Filter** section as follows. The value must be a Regex pattern used to filter out ACW whose key name matches the pattern.

```
...
[filter]
acw_custom_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/2/library/re.html.

4. Add a new option called **acw_custom_data_filter_exception** to this section as follows. The value must be a Regex pattern used to exclude ACW that should not be filtered out (for example, like GRECORD_PARTITIONS).

```
...
[filter]
acw_custom_data_filter = ^ORSI:     ;   (Note: this is the default value when the option
is not specified.)
acw_custom_data_filter_exception = ^GRECORD_PARTITIONS$     ;   (Note: this is the
default value when the option is not specified.)
...
```

For information about Regex patterns, refer to the Python's documentation found here: https://docs.python.org/2/library/re.html.

5. Restart the Recording Processor.

## Configuring SSL for Recording Processor

To configure SSL:

### Configure HTTPS on the Primary Recording Processor Server

1. Make sure pyOpenSSL is installed.
2. Create a self-signed certificate and private key for the Recording Processor host. For example, on RHEL run: `openssl req -new -x509 -days 1024 -nodes -out cert228.pem -keyout cert228key.pem`
3. In the `rp_server` section of the Recording Processor's configuration file, set the following parameters:
   - `ssl_certificate`—To point to the certificate PEM file. For example, `ssl_certificate=cert228.pem`.
   - `ssl_private_key`—To point to the private key file. For example, `cert228.pem`.
4. Give the self-signed certificate PEM file to any MCP client that needs to validate the certificate during the SSL handshake. See the "Enable Secure Communication" section Genesys Voice Platform 8.5 User's Guide.

5. Restart Recording Processor.

### Configure the HTTPS connection to Interaction Recording Web Services (Web Services)

1. Set up HTTPS on Interaction Recording Web Services (or Web Services if you're using version 8.5.210.02 or earlier). See the Genesys Security Deployment Guide.
2. Get the corresponding certificate for the Interaction Recording Web Services (Web Services) server. Set the caCertificate option in your Interaction Recording Web Services application (see caCertificate if you're using a Web Services application).
3. In the **[htcc]** section of the Recording Processor configuration file, set **base_uri** parameter to use `https`.
4. In the **[client]** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA certificate file. See the `<Python27 install directory>\Lib\site-packages\httplib2\cacerts.txt` file for an example.

> **Important**
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

### Configure the HTTPS connection to SpeechMiner

1. Set up HTTPS on SpeechMiner.See the Genesys Security Deployment Guide.
2. Set the **disable_ssl_certificate_validation** parameter in the **[speechminer]** section of the Recording Processor configuration to a value of 1.
3. Using Genesys Administrator Extension on the `Recording` tab of the IVR Profile, modify the `SpeechMiner Interaction Receiver` field use `https` as the protocol in the URL.
4. In the **[client]** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA cert file. See the `<Python27 install directory>\Lib\site-packages\httplib2\cacerts.txt` file for an example.

> **Important**
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

### Configure the HTTPS connection from the backup Recording Processor to the primary Recording Processor

1. Configure HTTPS on the primary Recording Processor.
2. Get the corresponding PEM certificate for the Web Services server.

3. In the **[processing]** section of the Recording Processor configuration file, set the **post_uri** parameter to use `https` as the protocol in the URL.
4. In the **client** section, set the **certs** parameter to point to the file that contains the certificate (see previous step). Or, point to an existing CA certificate file after copying the content of the new certificate into the existing CA certificate file. See the `<Python27 install directory>\Lib\site-packages\httplib2\cacerts.txt` file for an example.

> **Important**
> If there are multiple client connections using HTTPS, use a single CA cert file with all the certificates listed.

## Configure the HTTPS on the backup Recording Processor Server

Follow the same procedure used for the Primary Recording Processor Server using a new certificate and private key for the Backup Recording Processor's server.

## Configuring the IVR Profile

Using Genesys Administrator Extension, configure the following parameters on the **Recording** tab of the IVR Profile:

1. **Recording Processor URI**—The URI that the Media Control Platform (MCP) uses to post the metadata of the audio recording after the recording is complete. For example, `http:// <Recording Processor Host>/api/contact-centers/<Contact Cente Domain Name>/recordings/`.

> **Important**
> The value for the URI must always end with a forward slash (/).

2. SpeechMiner Interaction Receiver—The URL that points to the SpeechMiner service responsible for accepting metadata from the RPS for this profile.
3. SpeechMiner Interaction Receiver Authorization Header—The authorization information required to connect to the SpeechMiner service used by the RPS. For example, `<SpeechMiner Webserver Username>:<SpeechMiner Webserver Password>`.

For more information, see the Configuring GVP.

## Configuring the Recording Processor Using Genesys Administrator Extension (Optional)

The Recording Processor uses a configuration file instead of a specific application object in Configuration Server. However, it is possible to configure the Recording Processor as a "third-party server" application enabling Genesys Administrator Extension to monitor, start, and stop the process.

The following steps describe how to setup Recording Processor as a "third party server" application in Genesys Administrator Extension. For more information, see the "Using the Management Layer" section of the Framework 8.5.1 Management Layer User's Guide

Configuring RPS to Start/Stop via LCA using Genesys Administrator Extension:

1. Install and deploy the latest RPS.
2. Make sure that the Local Control Agent (LCA) is running.
3. Create a new application template in Genesys Administrator Extension called `Recording Processor Script` of type `Third Party Server`.
4. Create a new application (for example, `myRPS`) in Genesys Administrator Extension using this new application template.
5. Set the `Command Line` parameter (for example, C:\Python27\python.exe).
6. Set the `Host` parameter in the application's server info to the correct Host object.
7. Set the `Working Directory` parameter to the `<Recording Processor Install Directory>\rp` directory. For example, `/opt/genesys/Recording_Processor_Script_8.5/rp/`.
8. Set the `Command Line Arguments` parameter to the appropriate values. For example, `recording_process.py --config-file=/opt/genesys/Recording_Processor_Script_8.5/rp/rpconfig.cfg`.
   Refer to the Starting the Recording Processor Script section for additional command line parameters
9. Make sure that LCA has permission to read and write to the Recording Processor installation directory and Recording Processor log directory.
10. Save the configuration changes.
11. Ensure that the Configuration Server parameters in the Recording Processor configuration file are set appropriately. Refer to **Configure the Configuration Server Connection** tab on this page.

> **Important**
>
> The Recording Processor does not support configuration through Genesys Administrator Extension. Configuration is acquired using a local configuration file.

For more information about the RPS options, see Genesys Interaction Recording Options Reference.

## Starting the Recording Processor Script

To launch the RPS, run the following command from the `<Recording Processor Install Directory>`:

```
<python executable> recording_process.py --config-file=rpconfig.cfg
```

Use the following command line when you want to run multiple instances of RPS on the same machine:

```
<python executable> recording_process.py --config-file=rpconfig.cfg --id=1 --port=8889
```

For each RPS instance, assign a unique id (--id parameter) and port number (--port).

### Important

- `--port` defines the server port opened by the RPS process.
- `--id` represents the suffix of the:
  - application_name in the configuration file. For example, if application_name is defined in the configuration file as **`RecordingProcessorScript`** and `--id 2` is specified in the command line, then the application object named **`RecordingProcessorScript_2`** will be used to start the program.
  - log files
  - metadata json files created in the failed folder
  - database file created by the process

By default the RPS log file is stored in the working directory. This can be changed by specifying a preexisting folder in the `logfile_path` parameter in the log file section of the configuration file. For example:
`logfile_path = C:\logs\recordingProcessor`