Genesys Security Deployment Guide

Outbound Contact Support for GDPR

5/5/2025

Contents

Outbound Contact Support for GDPR

This page describes product-specific aspects of Outbound Contact support for the European Union's General Data Protection Regulation (GDPR) in premise deployments. For general information about Genesys support for GDPR compliance, see General Data Protection Regulation.

> **Warning**
> Disclaimer: The information contained here is not considered final. This document will be updated with additional technical information.

Logging

Sensitive data masking in logs

Behavior of the OCS component in relation to handling PII data in the logs should be configured with a set of options defined in sections **[log-filter]** and **[log-filter-data]**.

The **[log-filter]** section defines the default treatment of filtering data in log output. It defines the treatment of all KV pairs in the User Data, Extensions, and Reasons attributes of the log, and also defines the behavior of selected call handling (such as T-Servers) and reporting applications when processing call related data.

The **[log-filter-data]** section defines the treatment of specific KV pairs in the User Data, Extensions, and Reasons attributes of the log. It overrides the general settings in the **[log-filter]** section. This section defines how the set of keys in User Data, Extensions, and Reason should be handled when they are printed out into the log files.

Refer to Common Configuration Options section for full details on sensitive data masking in logs.

Log rotation

Logging should be configured with the “expire” option that defines if the log files will expire, and if so, the maximum number of days before log files are deleted.

Sensitive data masking in log files and log retention implements the Privacy by Design GDPR requirement.

Refer to Common Configuration Options section for full details on log rotation.

Handling PII data

The PII data with which OCS operates consists of phone numbers, company names, and any user data that can be dynamically formed based on the business process. A company that implements the structure of the user data based on its business process should care about handling this user data. OCS stores this PII data in databases in the form of calling lists; the PII data could also appear in different log files of the OCS component itself.

The following table summarizes OCS sources which could contain PII data:

Source	Form of storage	PII data
Calling List(s)	Table in the relational database	Potentially any type of PII data, stored in user-defined fields
Application Logs	Flat file(s)	Potentially any type of PII data
Audit Trail Logs	Flat file(s)	Phone numbers
Do Not Call List(s)	Table in the relational database	Phone numbers, Customer IDs
GSW Request Log(s)	Table in the relational database	Phone numbers
Record History Log(s)	Flat files(s)	Potentially any type of PII data, stored in user-defined fields

Databases

OCS uses databases for two types of entities--Calling and Do Not Calling Lists, and GSW Request Logs. Database administrators should follow general rules for maintaining GDPR-compliant databases. The general approach is as follows.

- Design data location – operating systems, primary and backup nodes.
- Design data access – limiting personal data access to as few as possible persons and roles.
- Design data storage – different storage systems provide number of mechanisms allows to store sensitive data securely. It could be full or part data encryption, secure protocols, and so on.

All these items implement Privacy By Design GDP requirement for Calling List.

If there are databases that are not encrypted and contains PII data, these databases should be reviewed for sensitive data and corresponding records should be either modified or removed.

OCS database:

1. OCS Calling Lists – these contain phone numbers and user-defined fields which could contain PII
2. Do Not Call lists
3. GSW Request Logs

Handling Requests

Find / Export PII data

- Find/Export all records in the Calling List where phone number equals given:

```
SELECT * FROM <cl_table_name> WHERE contact_info LIKE '<phone number>'
```

- Find/Export all records in Calling List where user_field contains given identifier

```
SELECT * FROM <cl_table_name> WHERE <user_field> LIKE '%<identifier>% '
```

- Find/Export data in Do Not Call Lists where phone number equals given.

```
SELECT * FROM <dnc_table_name> WHERE phone LIKE '<phone number>'
```

- Find/Export data in Do Not Call Lists where Customer ID equals given.

```
SELECT * FROM <dnc_table_name> WHERE customer_id LIKE '<identifier> '
```

- Find/Export data in GSW Request Log where phone number equals given.

```
SELECT * FROM <rl_table_name> WHERE phone LIKE '<phone number>'
```

Edit PII data For archived Calling Lists, it can be done using SQL queries.

For example, the following SQL statement will mask phone numbers in the GSW Request Log where phone number matches a given number.

```
UPDATE <rl_table_name> SET phone = '***' WHERE phone LIKE '<phone number>'
```

A similar SQL statement could be used for masking PII data in the Calling List:

```
UPDATE <cl_table_name> SET contact_info = '***', <user_field> = '***' WHERE contact_info LIKE
'<phone number>'
```

Note, it is not recommended to update records that are retrieved or may be retrieved by OCS. To avoid updating such records, add the following clause to the WHERE part of the SQL statement above:

```
record_status NOT IN (2)
```

SQL Statement for Do Not Call List:

```
UPDATE <dnc_table_name> SET phone = '***', customer_id = '***' WHERE phone LIKE '<phone
number>'
```

Note, updates in the Do Not Call List table will not update data in the OCS memory immediately. This happen only when the Do Not Call List table is re-read by OCS (refer to https://docs.genesys.com/Documentation/OU/8.1.5/Dep/CallingLists#Rereading_of_the_Do-Not-Call_List).

Delete PII dataSQL query example for individual entries deletion based on phone number or unique ID stored in the user-defined field:

```
DELETE FROM <cl_table_name> WHERE contact_info LIKE '<phone number>'

DELETE FROM <cl_table_name> WHERE <user_field> LIKE '%<identifier>% '
```

OCS Log files

All log files should be checked to determine if they contain any PII data. If any is found, it should be

either masked or removed from the files.

Log files:

- Main OCS and OCS HTTP proxy logs
- OCS Audit Trail logs
- Record History logs

Handling Log Files

Find / Export data in log files

To find and/or export PII data in a log file, simple console utilities like grep can be used.

For example, using the grep utility with regexp request for a log file will find all strings that contains a given string, such as `SocialSecurityNumber: 123456789` that is not masked, such as `SocialSecurityNumber: grep -n -e "SocialSecurityNumber: \([0-9]\) " OCSLogFile.log`

The found data could be easily exported using redirecting of the output into the file instead of the standard output:

grep -n -e "SocialSecurityNumber: \([0-9]\) " OCSLogFile.log > ExportedData.txt

It implements Right of Access and Portability GDP requirement for log files.

Edit data in log files

To mask the PII data in the log files, find and edit procedures should be implemented for the log files. It can be done with some already-existing tools, some special tools designed for this purpose, or some general tools like SED in Linux based systems.

For example, using the SED utility with regexp request will update the log file in place and all strings like `SocialSecurityNumber: 123456789` will be changed to strings like `SocialSecurityNumber: ***`.

```
sed -i -e 's/SocialSecurityNumber: \([0-9]*\)/SocialSecurityNumber: ***/g'
OCSLogFile.log
```

Delete data from log files

Deleting of the PII data from the log files also should be implemented using find and edit procedures. It can also be done with special or general tools.

For example using the SED utility with regex request will completely remove the string where the pattern `SocialSecurityNumber: 123456789` has been found.

```
sed -i -e 's/^.*SocialSecurityNumber: \([0-9]*\).*$//g' tst.txt
```

The following example looks in all log files for the pattern `SocialSecurityNumber: 123456789` and deletes these files.

```
find ./ -iname "*.log" -exec grep -e "SocialSecurityNumber: \([0-9]\)" '{}' \;
-delete
```