Genesys Interaction Recording Solution Guide

Configuring Features

9/6/2025

Contents

Configuring Features

Review the sections below for more information about how to configure Interaction Recording Web Services to use the specified features.

Configuration for Voice Recordings

Interaction Recording Web Services requires a specific configuration for GIR ***call*** recordings to work correctly. The following sections describe how to configure Interaction Recording Web Services for call recordings.

Configuring the Interaction Recording Web Services Parameters

1. To support call recordings, it's important that you update the following settings in the serverSettings section of the **application.yaml** file:
   - undocumentedExternalApiUrl
   - createCallRecordingCF
   - crClusterName
   - crRegion
   - cryptoSecurityKey
   - webDAVMaxConnection
   - webDAVMaxTotalConnection
   - nodePath
   - recordingSettings, in particular **recordCryptoServerDecryptMaxConnection**, **recordCryptoServerDecryptMaxTotalConnection** and **recordCryptoServerDecryptSocketTimeout**
   - multiPartResolverMaxUploadSize
   - multiPartResolverMaxInMemorySize
   - backgroundScheduledMediaOperationsSettings, in particular **enableBackgroundScheduledMediaOperations** and **defaultBackupExportURI**
2. Determine the contact center ID for Interaction Recording Web Services using the following command with the ops username and password (`ops:ops`):

   ```
   {
   curl -u ops:ops http://<Interaction Recording Web Services Server>:<Interaction
   Recording Web Services port>/api/v2/ops/contact-centers; echo
   }
   ```

   Interaction Recording Web Services returns the following output:

   ```
   {"statusCode":0,"uris":["http://<Interaction Recording Web Services Server>:<Interaction
   ```

```
Recording Web Services port>/api/v2/ops/
contact-centers/<contact center ID (in hex format)>"]}
```

3. Using a text editor, create a new file called `add_voice_features` with the following content:

```
{
      "uris":[
       "/api/api-voice-recording",
       "/api/api-supervisor-recording",
       "schema-elasticsearch-v2-call-recording"
   ]
}
```

4. Execute the following command:

```
{
curl -u ops:ops -X POST -d @add_voice_features
 http://<Interaction Recording Web Services Server>:<Interaction Recording Web Services
Port>/api/v2/ops/contact-centers/<contact center ID (in hex format)>/features
 --header "Content-Type: application/json"; echo
}
```

Configuring the Storage Credentials for Interaction Recording Web Services

Enable Voice Recording

Start

1. Determine the contact center ID on Interaction Recording Web Services using the following command with the ops username and password (`ops:ops`):

```
{
curl -u ops:ops http://<Interaction Recording Web Services Server>:<Interaction
Recording Web Services Port>/api/v2/ops/contact-centers; echo
}
```

The following output is returned:

```
{"statusCode":0,"uris":["http://<Interaction Recording Web Services Server>:<Interaction
Recording Web Services Port>/api/v2/ops/contact-centers/<contact center ID (in hex
format)>"]}
```

> **Important**
>
> Use the `<contact center ID (in hex format)>` in all subsequent commands.

2. Using a text editor, create a new file called `create_table` with the following content:

```
{
"operationName":"createCRCF"
}
```

> **Important**
>
> You do not need to create the table manually when the **createCallRecording** option is set to `true` in the **application.yaml** file. The table will be automatically created by Interaction Recording Web Services (RWS).

3. Execute the following command:

```
{
curl -u ops:ops -X POST -d @create_table http://<Interaction Recording Web Services
Server>:<Interaction Recording Web Services Port>/api/v2/ops/
contact-centers/<contact center ID (in hex format)>/recordings
 --header "Content-Type: application/json"; echo
}
```

End

Enable Storage

Start

1. Using a text editor, create a new file called `recording_settings` with the following content:

```
{
   "store": [
     {
       "webDAV": {
         "userName": "user1",
         "password": "password1",
         "uri": "http://apache1/recordings"
       }
     },
     {
       "webDAV": {
         "userName": "user2",
         "password": "password2",
         "uri": "http://apache2/recordings"
                 }
       }
   ]
}
```

> **Important**
>
> The URI in `recording_settings` is case sensitive and must match the URI in the IVR Profile. For example:
>
> ```
> "uri": "http://GENESYSREC1/recordings"
> is not the same as
> "uri": "http://genesysrec1/recordings"
> ```

2. Execute the following command:

```
{
```

```
curl -u ops:ops -X POST -d @recording_settings
 http://<Interaction Recording Web Services Server>:<Interaction Recording Web Services
Port>/api/v2/ops/contact-centers/<contact center ID (in hex format)>/settings/call-
recordings
 --header "Content-Type: application/json"; echo
}
```

End

Configuring the Call Recording Audit Log

Interaction Recording Web Services provides an audit log for the following recording operations:

- Playback of the recording media file
- Deletion of the recording file

Complete the steps below to configure the audit log:

Start

1. Stop Interaction Recording Web Services using the following command:
   `sudo service gir stop`
2. Edit the **GWS_HOME/etc/logback.xml file** file and update the configuration to include INFO level messaging. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Example LOGBACK Configuration File
  http://logback.qos.ch/manual/configuration.html
  -->
<configuration scan="true">
  <appender name="RECORDING" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <filter class="ch.qos.logback.classic.filter.LevelFilter">
      <level>INFO</level>
      <onMatch>ACCEPT</onMatch>
      <onMismatch>DENY</onMismatch><!-- ACCEPT for printing log above INFO, DENY for
printing only INFO-->
    </filter>
    <file>${jetty.logs}/recording.log</file>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <fileNamePattern>${jetty.logs}/recording-%d{yyyy-MM-dd-HH}.gz</fileNamePattern>
      <maxHistory>720</maxHistory><!-- 1 Month -->
    </rollingPolicy>
    <encoder>
      <pattern>%d{MM/dd/yyyy HH:mm:ss.SSS, UTC} [%X{principal.name}] [%X{req.userAgent}]
[%X{req.remoteHost}] %X{req.requestURI} %msg%n</pattern>
    </encoder>
  </appender>
  <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>${jetty.logs}/cloud.log</file>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <!-- hourly rollover -->
      <fileNamePattern>${jetty.logs}/cloud-%d{yyyy-MM-dd-HH}.gz</fileNamePattern>
      <!-- keep 5 days' worth of history -->
      <maxHistory>120</maxHistory>
```

```
    </rollingPolicy>
    <encoder>
      <pattern>%d{MM/dd/yyyy HH:mm:ss.SSS, UTC} %-5level [%X{principal.name}]
[%X{session}] [%X{contactCenter}] [%thread] %X{req.requestURI} %X{req.queryString}
%logger{36} %msg%n</pattern>
    </encoder>
  </appender>
  <logger name="com.<domain>.cloud.v2.api.controllers.callrecording">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.<domain>.cloud.v2.api.tasks.callrecording">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.<domain>" level="WARN" />
  <logger name="com.<domain>.cloud" level="DEBUG" />
  <logger name="com.<domain>.cloud.rtreporting" level="WARN" />
  <logger name="com.<domain>.salesforce.security" level="INFO" />

  <root level="WARN">
    <appender-ref ref="FILE" />
  </root>

</configuration>
```

3. For MLM, create a **RECORDING** appender if it does not exist. For example:

```
<appender name="RECORDING" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <filter class="ch.qos.logback.classic.filter.LevelFilter">
      <level>INFO</level>
      <onMatch>ACCEPT</onMatch>
      <onMismatch>DENY</onMismatch><!-- ACCEPT for printing log above INFO, DENY for
printing only INFO-->
    </filter>
    <file>${jetty.logs}/recording.log</file>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <fileNamePattern>${jetty.logs}/recording-%d{yyyy-MM-dd}.gz</fileNamePattern>
      <maxHistory>720</maxHistory><!-- 1 Month -->
    </rollingPolicy>
    <encoder>
      <pattern>%d{MM/dd/yyyy HH:mm:ss.SSS, UTC} [%X{principal.name}] [%X{req.userAgent}]
[%X{req.remoteHost}] %X{req.requestURI} %msg%n</pattern>
    </encoder>
  </appender>
```

4. Add the following loggers for the **RECORDING** appender:

```
<logger name="com.genesyslab.cloud.v2.api.controllers.callrecording">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.genesyslab.cloud.v2.api.controllers.screenrecording">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.genesyslab.cloud.v2.api.tasks.callrecording">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.genesyslab.cloud.v2.api.tasks.interactionrecording">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.genesyslab.cloud.v2.api.tasks.screenrecording">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.genesyslab.cloud.v2.api.tasks.settings">
```

```
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.genesyslab.cloud.v2.media.scheduler">
    <appender-ref ref="RECORDING" />
  </logger>
  <logger name="com.genesyslab.cloud.v2.media.task">
    <appender-ref ref="RECORDING" />
  </logger>
```

For more information about Logback, see Logback configuration.

5. Start GIR using the following command:
   `sudo service gir start`
6. Review the audit log. Open the <LOG_PATH>/recording.log file, where <LOG_PATH> is the path parameter for the logging section in your application.yaml. By default, this is /var/log/jetty9. The following example shows that two recordings are requested for playback and deletion:

```
10/28/2013 15:46:03.203 [ops] [Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_0) AppleWebKit/
537.36 (KHTML, like Gecko)
Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/ops/contact-centers/46284f2f-
d615-4329-957a-f5341ed
fd5d7/recordings/recid0/play/2cb4ea04-f81d-44e8-83b6-1f4a63a1a659.mp3 Play media
[2cb4ea04-f81d-44e8-83b6-1
f4a63a1a659] of recording [recid0] from contact center [46284f2f-d615-4329-957a-
f5341edfd5d7] requested

10/28/2013 15:46:03.341 [ops] [Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_0)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/ops/contact-centers/46284f2f-
d615-4329-957a-f5341ed
fd5d7/recordings/recid0/play/2cb4ea04-f81d-44e8-83b6-1f4a63a1a659.mp3 Play media
[2cb4ea04-f81d-44e8-83b6-1
f4a63a1a659] of recording [recid0] from contact center [46284f2f-d615-4329-957a-
f5341edfd5d7] failed

10/28/2013 15:46:10.946 [ops] [Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_0)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/ops/contact-centers/46284f2f-
d615-4329-957a-f5341ed
fd5d7/recordings/recid1/play/2cb4ea04-f81d-44e8-83b6-1f4a63a1a658.mp3 Play media
[2cb4ea04-f81d-44e8-83b6-1
f4a63a1a658] of recording [recid1] from contact center [46284f2f-d615-4329-957a-
f5341edfd5d7] requested

10/28/2013 15:46:11.033 [ops] [Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_0)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/ops/contact-centers/46284f2f-
d615-4329-957a-f5341ed
fd5d7/recordings/recid1/play/2cb4ea04-f81d-44e8-83b6-1f4a63a1a658.mp3 Play media
[2cb4ea04-f81d-44e8-83b6-1
f4a63a1a658] of recording [recid1] from contact center [46284f2f-d615-4329-957a-
f5341edfd5d7] succeed

10/28/2013 15:46:52.179 [admin@genesyslab.com] [Mozilla/5.0 (Macintosh; Intel Mac OS X
10_9_0) AppleWebKit/537.36 (
KHTML, like Gecko) Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/recordings/
recid0 Delete metadata and media-files of call-recording is requested. contact-center
[46284f2f-d615-4329-957a-f5341edfd5d7], call-recording [recid0]

10/28/2013 15:46:52.216 [admin@genesyslab.com] [Mozilla/5.0 (Macintosh; Intel Mac OS X
10_9_0) AppleWebKit/537.36 (
KHTML, like Gecko) Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/recordings/
```

```
recid0 Delete metadata and media-files of call-recording failed. contact-center
[46284f2f-d615-4329-957a-f5341edfd5d7], call-recording [recid0]

10/28/2013 15:46:56.253 [admin@genesyslab.com] [Mozilla/5.0 (Macintosh; Intel Mac OS X
10_9_0) AppleWebKit/537.36 (
KHTML, like Gecko) Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/recordings/
recid1 Delete metadata of call-recording is requested. contact-center [46284f2f-
d615-4329-957a-f5341edfd5d7], call-recording [recid1]

10/28/2013 15:46:56.420 [admin@genesyslab.com] [Mozilla/5.0 (Macintosh; Intel Mac OS X
10_9_0) AppleWebKit/537.36 (
KHTML, like Gecko) Chrome/28.0.1500.71 Safari/537.36] [192.168.135.1] /api/v2/recordings/
recid1 Delete metadata of call-recording succeeded. contact-center [46284f2f-
d615-4329-957a-f5341edfd5d7], call-recording [recid1]
```

End

Configuring the API Thread Pool

Interaction Recording Web Services provides properties for the Call Recording API thread pool by configuring the **hystrix.properties** file.

The following table describes the parameters required to set the API thread pool.

Property/API Name	Thread Pool Name	Description
hystrix.command.[API Name]. execution.isolation.thread. timeoutInMilliseconds	N/A	The hystrix timeout. The default value is set to 6000.
hystrix.threadpool.[API Pool Name] .coreSize	N/A	The thread pool size. The default value is set to 10.
RecordingOperationApiTaskV2	ApiOperationPool	The call or screen recording operation.
CreateCallRecordingApiTaskV2	ApiCreatePool	Create call recording.
DeleteCallRecordingApiTaskV2	ApiDeletePool	Delete call recording.
GetCallRecordingApiTaskV2	ApiGetPool	Get call recording metadata.
GetCallRecordingCFInfoApiTaskV2	ApiGetPool	Get call recording CF Information.
GetCallRecordingMediaApiTaskV2	ApiGetPool	Streaming call recording media.
QueryCallRecordingApiTaskV2	ApiQueryPool	Query call recording metadata.

For more information about the Call Recording API, see the Genesys Interaction Recording API Reference.

Configuration for Screen Recordings

As with call recordings, Interaction Recording Web Services requires a specific configuration for GIR ***screen*** recordings to work correctly. The following sections describe how to configure Interaction Recording Web Services for screen recordings.

Configuring the Interaction Recording Web Services Parameters

Complete the steps below to support screen recordings:

Start

1. For **Cassandra 2.2**, Update the following settings in the serverSettings section of the **application.yaml** file. Your configuration should look something like this:

```
crossOriginSettings:
    corsFilterCacheTimeToLive: 120
    allowedOrigins: <Interaction Recording Web Services Servers>,<SpeechMiner Web
Servers>
    allowedMethods: GET,POST,PUT,DELETE,OPTIONS
    allowedHeaders: "X-Requested-With,Content-
Type,Accept,Origin,Cookie,authorization,ssid,surl,ContactCenterId,Range"
    allowCredentials: true
screenRecordingSettings:
    screenRecordingEServicesEnabled: true
    screenRecordingVoiceEnabled: true
screenRecordingConnectionReportingSettings:
  reportingEnabled: true
  createReportingCF: true
multiPartResolverMaxUploadSize: 536870912
multiPartResolverMaxInMemorySize: 67108864
```

   For **Cassandra 4.1**, Update the following settings in the serverSettings section of the **application.yaml** file. Your configuration should look something like this:

```
crossOriginSettings:
    corsFilterCacheTimeToLive: 120
    allowedOrigins: <Interaction Recording Web Services Servers>,<SpeechMiner Web
Servers>
    allowedMethods: GET,POST,PUT,DELETE,OPTIONS
    allowedHeaders: "X-Requested-With,Content-
Type,Accept,Origin,Cookie,authorization,ssid,surl,ContactCenterId,Range"
    allowCredentials: true
screenRecordingSettings:
    screenRecordingEServicesEnabled: true
    screenRecordingVoiceEnabled: true
screenRecordingConnectionReportingSettings:
  reportingEnabled: false
  createReportingCF: false
multiPartResolverMaxUploadSize: 536870912
multiPartResolverMaxInMemorySize: 67108864
```

   Make the following changes to the example above:

   - Change `<Interaction Recording Web Services Servers>` and `<SpeechMiner Web Servers>` to the HTTP/HTTPS addresses of the Interaction Recording Web Services instances and SpeechMiner Web Servers.

- multiPartResolverMaxUploadSize controls the maximum allowed size (in bytes) for a screen recording video file that can be uploaded to Interaction Recording Web Services. This parameter should be aligned with `maxDurationMinutes`, so if you change its value, ensure that you also consider the `maxDurationMinutes` value specified within the *Advanced Configuration for the Screen Recording Service* section in the Deploying the Screen Recording Service - Advanced Configuration page. The maximum size of a file that can be uploaded by the Screen Recording Service must be less than or equal to the `multiPartResolverMaxUploadSize`.

2. Determine the contact center ID on Interaction Recording Web Services using the following command with the ops username and password (`ops:ops`):

```
{
curl -u ops:ops http://<Interaction Recording Web Services Server>:<Interaction
Recording Web Services port>/api/v2/ops/contact-centers; echo
}
```

Interaction Recording Web Services returns the following output:

```
{"statusCode":0,"uris":["http://<Interaction Recording Web Services Server>:<Interaction
Recording Web Services port>/api/v2/ops/
contact-centers/<contact center ID (in hex format)>"]}
```

3. For **Cassandra 2.2**, Using a text editor, create a new file called `add_screen_features` with the following content:

```
{
      "uris":[
       "/api/api-voice-screenrecording",
       "/api/api-multimedia-screenrecording",
       "/api/api-screenrecording-connection-reporting",
       "schema-elasticsearch-v2-screen-recording"
   ]
}
```

For **Cassandra 4.1**, Using a text editor, create a new file called `add_screen_features` with the following content:

```
{
      "uris":[
       "/api/api-voice-screenrecording",
       "/api/api-multimedia-screenrecording",
       "schema-elasticsearch-v2-screen-recording"
   ]
}
```

4. Execute the following command:

```
{
curl -u ops:ops -X POST -d @add_screen_features
 http://<Interaction Recording Web Services Server>:<Interaction Recording Web Services
Port>/api/v2/ops/contact-centers/<contact center ID (in hex format)>/features
 --header "Content-Type: application/json"; echo
}
```

- Use the **api-voice-screenrecording** parameter for voice interactions, and use the **api-multimedia-screenrecording** parameter for non-voice interactions.
- Use the **api-screenrecording-connection-reporting** parameter to enable the collection of information about Screen Recording Services client connections for the contact center. **This feature is not applicable for Cassandra 4.1.**
- If you wish to direct the SpeechMiner UI to Interaction Recording Web Services instead of Recording Crypto Server for decryption of screen recordings, add the **api-recordings-decryption-proxying** parameter to the list of features enabled for the contact center above. Note that this requires

additional configuration.

5. Using a text editor, create a new file called `create_stats_table`, with the following content:**This step is not applicable for Cassandra 4.1.**

```
{
  "operationName":"CreateReportingCFs"
}
```

6. Execute the following command:**This step is not applicable for Cassandra 4.1.**

```
{
curl -u ops:ops -X POST -d @create_stats_table http://<Interaction Recording Web
Services Server>:<Interaction Recording Web Services Port>/api/v2/ops/contact-
centers/<contact center ID (in hex format)>/screen-recording-connections --header
"Content-Type: application/json"; echo
}
```

End

Configuring the Storage Credentials for Interaction Recording Web Services

Complete the steps below to configure storage credentials for Interaction Recording Web Services.

Start

1. Determine the contact center ID on Interaction Recording Web Services using the following command with the ops username and password (`ops:ops`):

```
{
curl -u ops:ops http://<Interaction Recording Web Services Server>:<Interaction
Recording Web Services port>/api/v2/ops/contact-centers; echo
}
```

Interaction Recording Web Services returns the following output:

```
{"statusCode":0,"uris":["http://<Interaction Recording Web Services Server>:<Interaction
Recording Web Services port>/api/v2/ops/
contact-centers/<contact center ID (in hex format)>"]}
```

> **Important**
>
> Use the `<contact center ID (in hex format)>` construction in all subsequent commands.

2. Using a text editor, create a new file called `create_table`, with the following content:

```
{
"operationName":"createCRCF"
}
```

3. Execute the following command:

```
{
```

```
curl -u ops:ops -X POST -d @create_table http:// <Interaction Recording Web Services
Server>:<Interaction Recording Web Services Port>/api/v2/ops/
contact-centers/<contact center ID (in hex format)>/screen-recordings
 --header "Content-Type: application/json"; echo
}
```

4. Enable storage for a single or multiple locations:

> **Important**
>
> Within the storage settings, the same location can be specified multiple times if you have inactive ("active": false) settings specified as well as "active": true. However, you must ensure that for a specific location, only one value has "active": true set. For additional information about storage settings, refer to Interaction Recording Web Services (Web Services) Group Settings. See the **Property Descriptions** section for details about the supported property values.

- For a **single** location:

a. Using a text editor, create the create_single_location file:

```
{
    "name":"storage",
    "location": "/",
    "value":[
        {
            "storageType": "webDAV",
            "active": true,
            "credential":
            {
                "userName": "<webdav user>",
                "password": "<webdav password>",
                "storagePath": "<webdav uri>"
            }
        }
    ]
}
```

> **Important**
>
> Replace <webdav user>, <webdav password>, <webdav uri> with the appropriate values.

b. Execute the following command:

```
{
curl -u ops:ops -X POST -d @create_single_location http:// <Interaction Recording Web
Services Server>:<Interaction Recording Web Services Port>/api/v2/ops
/contact-centers/<contact center ID (in hex format)>/settings/screen-recording
--header "Content-Type: application/json"; echo
}
```

- For **multiple** locations:

a. Using a text editor, create the create_first_location file:

```
{
    "name":"storage",
```

```
    "location": "<node_location>",
    "value":[
        {
            "storageType": "webDAV",
            "active": true,
            "credential":
            {
                "userName": "<webdav user>",
                "password": "<webdav password>",
                "storagePath": "<webdav uri>"
            }
        }
    ]
}
```

b. Execute the following command:

```
{
curl -u ops:ops -X POST -d @create_first_location http://<Interaction Recording
Web Services Server>:<Interaction Recording Web Services Port>/api/v2/ops
/contact-centers/<contact center ID (in hex format)>/settings/screen-recording
 --header "Content-Type: application/json"; echo
}
```

> **Important**
>
> Replace <node_location>, <webdav user>, <webdav password>, <webdav uri> with the appropriate values. The values for the <node_location> are similar to the nodePath settings in the **application.yaml** file, but allow a hierarchical representation. For example, an Interaction Recording Web Services node uses a storage setting with a location of "/US" in the nodePath set to "/US/AK" or "/US/HI".
>
> For more information on hierarchical location setting, go [[1]]

c. Repeat steps a and b for each location required.

End

For more information on the properties of this settings group, see Interaction Recording Web Services Settings Groups.

Configuring the API Thread Pool

Interaction Recording Web Services provides properties for the Call Recording API thread pool by configuring the **hystrix.properties** file.

The following table describes the parameters required to set the API thread pool.

Property/API Name	Thread Pool Name	Description
hystrix.command.[API Name]. execution.isolation.thread. timeoutInMilliseconds	N/A	The hystrix timeout. The default value is set to 6000.
hystrix.threadpool.[API Pool	N/A	The thread pool size. The default

Property/API Name	Thread Pool Name	Description
Name] .coreSize		value is set to 10.
RecordingOperationApiTaskV2	ApiOperationPool	The call or screen recording operation.
CreateScreenRecordingApiTaskV2	ApiUploadPool	Create screen recording
DeleteScreenRecordingMediaApiTaskV2	ApiDeletePool	Delete screen recording
GetScreenRecordingApiTaskV2	ApiGetPool	Get screen recording metadata
GetScreenRecordingMediaApiTaskV2	ApiStreamPool	Stream screen recording media
QueryScreenRecordingApiTaskV2	ApiQueryPool	Query screen recording metadata

For more information about the Call Recording API, see the Genesys Interaction Recording API Reference.