Reference Guide

AudioCodes Intuitive Human Communications for Chatbot Services

Voice.AI Gateway REST API

For AudioCodes Partners

Version 2.2

Notice

Information contained in this document is believed to be accurate and reliable at the time of printing. However, due to ongoing product improvements and revisions, AudioCodes cannot guarantee accuracy of printed material after the Date Published nor can it accept responsibility for errors or omissions. Updates to this document can be downloaded from https://www.audiocodes.com/library/technical-documents.

This document is subject to change without notice.

Date Published: November-16-2020

WEEE EU Directive

Pursuant to the WEEE EU Directive, electronic and electrical waste must not be disposed of with unsorted waste. Please contact your local recycling authority for disposal of this product.

Customer Support

Customer technical support and services are provided by AudioCodes or by an authorized AudioCodes Service Partner. For more information on how to buy technical support for AudioCodes products and for contact information, please visit our website at https://www.audiocodes.com/services-support/maintenance-and-support.

Documentation Feedback

AudioCodes continually strives to produce high quality documentation. If you have any comments (suggestions or errors) regarding this document, please fill out the Documentation Feedback form on our website at https://online.audiocodes.com/documentation-feedback.

Stay in the Loop with AudioCodes

Related Documentation

Document Name
Voice.AI Gateway Product Description
Voice.AI Gateway Security Guidelines

Document Revision Record

LTRT	Description
31020	Initial document release.

Table of Contents

1 Introduction

The REST API is designed for developers who wish to programmatically integrate the Voice.AI Gateway into their solution and for Administrators who wish to perform management and configuration tasks through automation scripts.

The REST API provides access to the resources through pre-defined URL paths. Each resource represents a specific configuration element, state object or maintenance action. The REST API uses standard HTTP/1.1 protocol. For security, all API calls are done over HTTPS.

Standard HTTP methods – GET, PUT, POST and DELETE – are used to read the resource's state and to create, update, or delete the resources (wherever applicable). Resource state is described in JSON format and included in the HTTP request or response bodies.

2 Client OAuth 2.0 Authentication

The REST API uses OAuth 2.0 authentication and authorization. OAuth 2.0 is the industry-standard protocol for authorization. OAuth 2.0 focuses on client developer simplicity while providing specific authorization flows for web applications, desktop applications, mobile phones, and living room devices.

Before the client can perform any actions on the Voice.AI Gateway, the client must send a token request to the Voice.AI Gateway (according to Section 4.4.2 of RFC 6749), containing the client ID, an client secret, and the requested OAuth 2.0 authorization scopes (see below). Permissions to the client are granted according to the client's scopes. The client ID, client secret, and scopes are configured for the client (user) by AudioCodes through the Voice.AI Gateway's web-based management interface. Once generated, this information must be provided to the client by AudioCodes.

Listed below are the different client scopes:

- **partner**: This scope includes all the below scopes (i.e., full permission).
- **actions:dialout:** Required for performing dialout actions (Outbound Calling feature).
- **config:bots:read:** Required for GET operations on bot configuration.
- **config:bots:write:** Required for POST, PUT or DELETE operations on bot configuration.
- **config:providers:read:** Required for GET operations on provider configuration.
- **config:providers:write:** Required for POST, PUT or DELETE operations on provider configuration.
- User-defined scopes for fine-grained authorization of specific actions, like dialout.

The client token request should look like this (where CLIENT_ID and SECRET are obtained from AudioCodes):

```
POST /oauth/token HTTP/1.1
Host: server.example.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded

client_id=CLIENT_ID
&client_secret=SECRET
&grant_type=client_credentials
&scope=config:bots:read+config:providers:read
```

The response to the client from Voice.AI Gateway looks like this:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
```

```
Pragma: no-cache

{
"access_token":"ACCESS_TOKEN",
"token_type":"Bearer",
"expires_in":3600,
"scope": "config:bots:read config:providers:read"
}
```

Your client application should use the access token for subsequent requests. When the token expires, your client application should obtain a new access token by the same method described above.

When performing subsequent requests, the client must include the token as a bearer token in the Authorization header. The Voice.AI Gateway checks that the token is valid and contains the scopes needed for the requested action. If authorization fails, the Voice.AI Gateway responds to the client according to Section 3.1 of RFC 6750. The client should handle such failures and, if needed, obtain a new access token.

- The access token might expire prior to the reported expiration time. Hence, the client application should be ready at any time to handle the "401 Unauthorized" response indicating such expiry.
- You can configure the expiration time for the access token, using the `oauthTokenExpirationSeconds` parameter.

- Example of a missing Authorization header:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer realm="voice-ai"
```

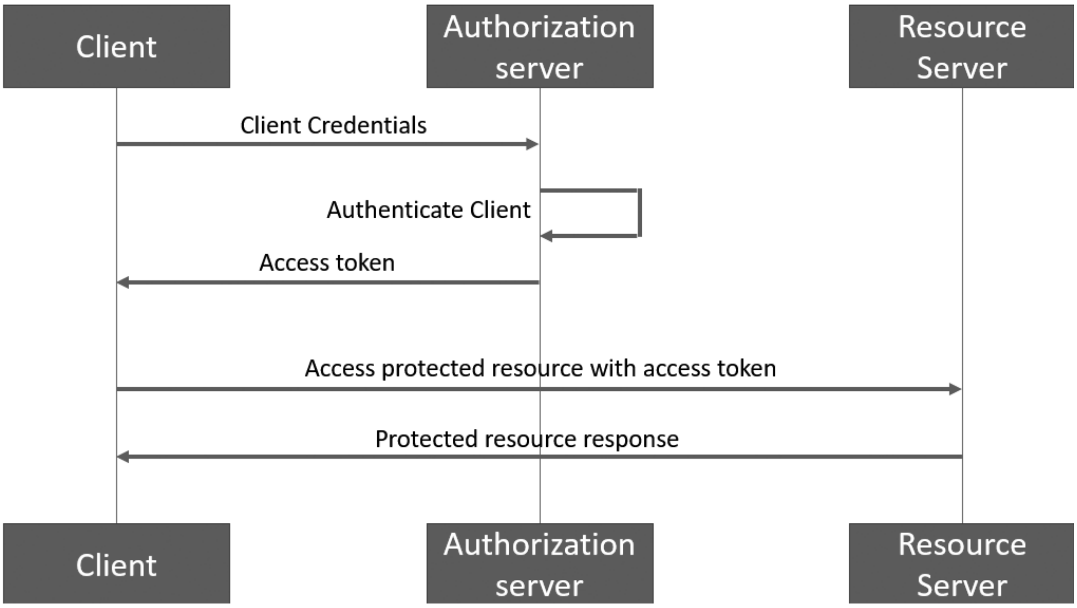
- Example of an expired token:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer realm="voice-ai",
 error="invalid_token",
 error_description="The access token expired"
```

- Example of insufficient scope:

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer realm="voice-ai",
 error="insufficient_scope",
 scope="config:bots:write"
```

The OAuth 2.0 authentication API flow is illustrated below:


Client
Authorization server
Resource Server
Client Credentials
Authenticate Client
Access token
Access protected resource with access token
Protected resource response
Client
Authorization server
Resource Server

3 REST API Commands

This section describes the REST API commands, which are used to configure the bots and providers (e.g., speech-to-text and text-to-speech providers) on the Voice.AI Gateway.

The REST API is done over HTTPS and supports the following HTTPS method requests:

- **POST:** This method creates bot(s) or provider(s).
- **PUT:** This method overwrites (updates) existing configuration of all or a specific bot or provider.
- **GET:** This method retrieves configuration of all or a specific bot or provider.
- **DELETE:** This method deletes a specific bot or provider.

The REST API requests are made from the REST client to the Voice.AI Gateway's (server), using the resource URI, `/api/v1/config/`. The REST requests can be done on the following REST resources:

REST Resource	HTTP Request	Description
`/api/v1/config/bots`	GET, POST and PUT	The requests apply to all bots.
`/api/v1/config/bots/{botName}`	GET, PUT and DELETE	The requests apply to a specific bot (specified by the `botName` parameter).
`/api/v1/config/providers`	GET, POST and PUT	The requests apply to all providers.
`/api/v1/config/providers/{providerName}`	GET, PUT and DELETE	The requests apply to a specific provider (specified by the `providerName` parameter).

The configuration parameters that are relevant for bots and providers are specified in Bots Section on page 21 and Providers Section on page 50, respectively.

⚠ Each bot should be configured with a provider.

All requests and responses contain a JSON body and with the appropriate 'Content-Type: application/json' header. All JSON bodies must be encoded with UTF-8.

When you call any of the REST APIs, one of the following standard HTTP status code is returned in the response header:

HTTP Status Codes	Description
200 OK	The request was successfully completed. A 200 status is returned for a successful GET, POST, PUT or DELETED method.
400 Bad Request	The request could not be processed because it contains missing or invalid information, such as a validation error on an input field, a missing required value, and so on.
401 Unauthorized	The request is not authorized. The OAuth 2.0 authentication token included with the request is missing or invalid.
404 Not Found	The request includes a resource URI that does not exist. For example, if a GET request is sent to retrieve information on a specific bot that doesn't exist.
409 Conflict	Indicates that a conflicting change has been detected during an attempt to modify a resource and that the request operation failed due to other concurrent operations. To resolve this issue, it's recommended to try the request again after a few seconds.

All Bots

This section describes APIs that apply to configuration of all bots.

GET api/v1/config/bots

Description

Retrieves configuration (items objects) of all bots.

REST Resource URL

```
/api/v1/config/bots
```

HTTP Method

```
GET
```

Media Type

`Accept: application/json`

Responses

Can return the following responses:

- 401: Authentication failure (missing OAuth token or invalid token)
- 200: Configuration of all the bots is included in the Content-Type header of the response, for example:

```
{
  "items": [
    {
      "name": "LondonTube",
      "provider": "my_azure",
      "displayName": "London Tube",
      "bargeIn": true,
      "url": "/api/v1/config/bots/LondonTube"
    },
    {
      "name": "Pizza",
      "provider": "my_aws",
      "sttProvider": "my_azure",
      "displayName": "Pizza",
      "providerBotName": "TalWantPizzaBot",
      "welcome": "Welcome to the Pizza service. I will try to do my best with your
pizza ordering. What would you like?",
      "sttDisablePunctuation": true,
      "dialoutAuthScope": "pizza-scope",
      "url": "/api/v1/config/bots/Pizza"
    }
  ]
}
```

⚠ For security, GET responses don't include the `credentials` field (of either the provider or bot). In other words, the client can configure the value for this field (using POST or PUT), but cannot read its current value.

POST /api/v1/config/bots

Description

Creates (adds) bot configuration.

> ⚠ If any of the provided bots already exist in the configuration, the POST request fails with a 400 error response.

REST Resource URL

```
/api/v1/config/bots
```

HTTP Method

```
POST
```

Media Type

`Accept: application/json`

Request Body Parameters

The request must contain the `name` parameter for each bot, as well as the `provider` and `displayName` parameters (unless `isBase` is `true`). Optionally, you can also configure additional parameters per bot, as described in Bots Section on page 21. Below is an example of a request body containing configuration of two bots, "LondonTube" and "Pizza":

```
[
    {
      "name": "LondonTube",
      "provider": "my_azure",
      "displayName": "London Tube",
      "bargeIn": true,
    },
    {
      "name": "Pizza",
      "provider": "my_aws",
      "sttProvider": "my_azure",
      "displayName": "Pizza",
      "providerBotName": "TalWantPizzaBot",
      "welcome": "Welcome to the Pizza service. I will try to do my best with your
pizza ordering. What would you like?",
      "sttDisablePunctuation": true,
      "dialoutAuthScope": "pizza-scope",
    }
]
```

Response

The response can be one of the following:

- 200: The specified bots were successfully added.
- 400: Validation error
- 401: Authentication failure (missing OAuth token or invalid token)
- 409: Operation failed due to other concurrent operations

For example:

```
{
"statusCode" : 200
}
```

PUT api/v1/config/bots

Description

Overwrites (replaces) all existing bot configuration with the new bot(s) configuration.

REST Resource URL

```
/api/v1/config/bots
```

HTTP Method

```
PUT
```

Media Type

`Accept: application/json`

Request Body (JSON)

The request must contain the `name` parameter for each bot, as well as the `provider` and `displayName` parameters (unless `isBase` is `true`). Optionally, you can also configure additional parameters per bot, as described in Bots Section on page 21. Below is an example of a request body containing configuration of two bots, "LondonTube" and "Pizza":

```
[
    {
      "name": "LondonTube",
      "provider": "my_azure",
      "displayName": "London Tube",
      "bargeIn": true,
    },
    {
      "name": "Pizza",
      "provider": "my_aws",
      "sttProvider": "my_azure",
      "displayName": "Pizza",
      "providerBotName": "TalWantPizzaBot",
      "welcome": "Welcome to the Pizza service. I will try to do my best with your
pizza ordering. What would you like?",
      "sttDisablePunctuation": true,
      "dialoutAuthScope": "pizza-scope",
    }
]
```

Response

The response can be one of the following:

- 200: Bot configuration was updated successfully
- 400: Validation error
- 401: Authentication failure (missing OAuth token or invalid token)
- 409: Operation failed due to other concurrent operations

For example:

```
{
"statusCode" : 200
}
```

Specific Bot

This section describes APIs that apply to configuration of a specific bot.

GET /api/v1/config/bots/{botName}

Description

Retrieves configuration (items objects) of a specific bot (`botName`).

REST Resource URL

```
/api/v1/config/bots/{botName}
```

HTTP Method

```
GET
```

Media Type

`Accept: application/json`

Responses

The response can be one of the following:

- 200: Configuration of the specified bot was successfully retrieved. Below shows an example of a response from a GET request for the resource /api/v1/config/bots/LondonTube:

```
{
  "name": "LondonTube",
  "provider": "my_azure",
  "displayName": "London Tube",
  "bargeIn": true,
  "url": "/api/v1/config/bots/LondonTube"
}
```

- 401: Authentication failure (missing OAuth token or invalid token)
- 404: Not found

⚠ For security, GET responses don't include the `credentials` field (of either the provider or bot). In other words, the client can configure the value for this field (using POST or PUT), but cannot read its current value.

PUT /api/v1/config/bots/{botName}

Description

Creates or updates configuration of a specific bot (`botName`). A PUT request on an existing bot overwrites the existing bot configuration with the new configuration.

REST Resource URL

```
/api/v1/config/bots/{botName}
```

HTTP Method

```
PUT
```

Media Type

`Accept: application/json`

Request Body Parameters

The request must contain the `name` parameter for the bot. The `name` parameter must correspond to the `botName` provided in the URL. Optionally, you can also configure additional parameters for the bot, as described in Bots Section on page 21. Below is an example of a request body containing a configuration of a bot "LondonTube":

```
{
  "name": "LondonTube",
  "provider": "my_azure",
  "displayName": "London Tube",
  "bargeIn": true,
}
```

Responses

The response can be one of the following:

- 200: Configuration of the specified bot was successfully updated
- 400: Validation error
- 401: Authentication failure (missing OAuth token or invalid token)
- 409: Operation failed due to other concurrent operation

For example:

```
{
"statusCode" : 200
}
```

DELETE /api/v1/config/bots/{botName}

Description

Deletes configuration of a specific bot (`botName`).

REST Resource URL

```
/api/v1/config/bots/{botName}
```

HTTP Method

```
DELETE
```

Media Type

`Accept: application/json`

Response

The response can be one of the following:

- 200: Configuration of specified bot was successfully deleted
- 401: Authentication failure (missing OAuth token or invalid token)
- 404: Not found
- 409: Conflict occured

For example:

```
{
"statusCode" : 200
}
```

All Providers

This section describes APIs that apply to configuration of all providers.

GET /api/v1/config/providers

Description

Retrieves all providers configuration (items objects).

REST Resource URL

```
/api/v1/config/providers
```

HTTP Method

```
GET
```

Media Type

`Accept: application/json`

Request Body

None.

Responses

- 200: Array of all the providers, for example:

```
{
  "items": [
    {
      "name": "my_aws",
      "type": "aws",
      "region": "us-west-2",
      "url": "/api/v1/config/providers/my_aws"
    },
    {
      "name": "my_azure",
      "type": "azure",
      "region": "westus2",
      "url": "/api/v1/config/providers/my_azure"
    }
]
}
```

- 401: Authentication failure (missing OAuth token or invalid token)

⚠ For security, GET responses don't include the `credentials` field (of either the provider or bot). In other words, the client can configure the value for this field (using POST or PUT), but cannot read its current value.

POST /api/v1/config/providers

Description

Creates (adds) configuration for providers.

REST Resource URL

```
/api/v1/config/providers
```

HTTP Method

```
POST
```

Media Type

`Accept: application/json`

Request Body (JSON)

The request must contain the `name` parameter for each provider, as well as the `provider` and `displayName` parameters (unless `isBase` is `true`). Optionally, you can also configure additional parameters for each provider, as described in Bots Section on page 21. Below is an example of a request body containing configuration for two providers, "my_aws" and "my_azure":

```
[
    {
      "name": "my_aws",
      "type": "aws",
      "region": "us-west-2",
    },
    {
      "name": "my_azure",
      "type": "azure",
      "region": "westus2",
    }
]
```

Response

The response can be one of the following:

- 200: The specified providers were successfully added
- 400: Validation error
- 401: Authentication failure (missing OAuth token or invalid token)
- 409: Operation failed due to other concurrent operations

For example:

```
{
"statusCode" : 200
}
```

PUT /api/v1/config/providers

Description

Overwrites all providers configuration.

REST Resource URL

```
/api/v1/config/providers
```

HTTP Method

```
PUT
```

Media Type

`Accept: application/json`

Request Body Parameters

The request must contain the `name` parameter for each provider, as well as the `provider` and `displayName` parameters (unless `isBase` is `true`). Optionally, you can also configure additional parameters for each provider, as described in Bots Section on page 21. Below is an example of a request body containing configuration for two providers, "my_aws" and "my_azure":

```
[
    {
      "name": "my_aws",
      "type": "aws",
      "region": "us-west-2",
    },
    {
      "name": "my_azure",
      "type": "azure",
      "region": "westus2",
    }
]
```

Response

The response can be one of the following:

- 200: Providers configuration was updated successfully
- 400: Validation error
- 401: Authentication failure (missing OAuth token or invalid token)
- 409: Operation failed due to other concurrent operations

For example:

```
{
"statusCode" : 200
}
```

Specific Provider

This section describes APIs that apply to configuration of a specific provider.

GET /api/v1/config/providers/{providerName}

Description

Retrieves configuration (items objects) of a specific provider (`providerName`).

REST Resource URL

```
/api/v1/config/providers/{providerName}
```

HTTP Method

```
GET
```

Responses

The response can be one of the following:

- 200: Configuration of the specified provider was successfully retrieved. Below shows an example of a response from a GET request for the resource /api/v1/config/providers/my_aws:

```
{
  "name": "my_aws",
  "type": "aws",
  "region": "us-west-2",
  "url": "/api/v1/config/providers/my_aws"
}
```

- 401: Authentication failure (missing OAuth token or invalid token)
- 404: Not found

> ⚠ For security, GET responses don't include the `credentials` field (of either the provider or bot). In other words, the client can configure the value for this field (using POST or PUT), but cannot read its current value.

PUT /api/v1/config/providers/{providerName}

Description

Creates or updates configuration of a specific provider (`providerName`). A PUT request on an existing provider overwrites the existing provider configuration with the new configuration.

REST Resource URL

```
/api/v1/config/providers/{providerName}
```

HTTP Method

```
PUT
```

Request Body Parameters

The request must contain the `name` parameter of the provider. The `name` parameter must correspond to the `botName` provided in the URL. Optionally, you can also configure additional parameters for the provider, as described in Bots Section on page 21. Below is an example of a request body containing configuration of a provider "my_aws":

```
{
  "name": "my_aws",
  "type": "aws",
  "region": "us-west-2",
}
```

Responses

The response can be one of the following:

- 200: Configuration of the specified provider was successfully updated
- 400: Validation error
- 401: Authentication failure (missing OAuth token or invalid token)
- 409: Operation failed due to other concurrent operation

```
{
"statusCode" : 200
}
```

DELETE /api/v1/config/providers/{providerName}

Description

Deletes configuration of a specific provider (`providerName`).

REST Resource URL

```
/api/v1/config/providers/{providerName}
```

HTTP Method

```
DELETE
```

Response

The response can be one of the following:

- 200: Configuration of specified provider was successfully deleted
- 401: Authentication failure (missing OAuth token or invalid token)
- 404: Not found

- 409: Conflict occured

For example:

```
{
"statusCode" : 200
}
```

4 Bots Section

This section describes the bot parameters. All these parameters can be retrieved or set using the REST API (or using the Voice.AI Connector configuration file). Both the parameters that can only be configured (controlled) by the Administrator (see Parameters Controlled Only by Administrator below) and the parameters that can also be controlled by the bot (see Parameters Controlled Also by Bot on page 29) are configured together on the same JSON object.

Each bot is identified by the `name` parameter, which must be unique in the Voice.AI Gateway configuration.

Parameters Controlled Only by Administrator

These parameters are controlled only by the Voice.AI Gateway administrator (configured in the Configuration file or through REST API).

Table 4-1: Description of Bots Section Parameters (Controlled by Voice.AI Gateway Administrator)

Parameter	Type	Description
`isBase`	Boolean	Defines this bot's configuration as the "base" configuration. ■ `true` ■ `false` (Default) **Note:** ■ The "base" bot is **not** a real deployed bot (and cannot be used as a bot), but is only a template that allows other bots to inherit its configuration. ■ The following parameters should not be configured for the "base" bot: ✔ `extends` ✔ `provider` ✔ `displayName` ■ Multiple "base" bots can be configured.
`extends`	String	Specifies the "base" configuration (configured by the `isBase` parameter) to use for the current bot. The value of the parameter is the value of the `name` parameter of the "base" bot.

Parameter	Type	Description
		Note: ■ The bot inherits all the parameters from the "base" configuration, except the following: ✔ `name` ✔ `isBase` ■ If a parameter appears on both the "base" bot and the current bot configuration, the parameter value of the current bot configuration takes precedence. ■ The parameter must not be configured for the "base" bot.
`dialoutAuthScope`	String	Defines the OAauth 2.0 scope that is authorized to perform dialout (outbound calling) with the specific bot. Dialout clients who want to perform dialout on this App must have this scope defined in the Users table of the Voice.AI Gateway Web-based management user interface. Leaving this field empty (default) prevents (disables) the dialout feature for this bot.
`connectOnPrompt`	Boolean	Enables the Voice.AI Connector to wait for the first message from the bot before sending the Connect (200 OK) message to the user. ■ `true` (Default) ■ `false`
`customerProjectId`	String	Defines the Project ID of the bot. This is needed for connecting to the one-click Google Dialogflow application. **Note:** The parameter is applicable only to the One-Click Google Dialogflow application.
`displayName`	String	Defines the name of the bot that is displayed for easy identification in the Voice.AI Connector's Web user interface. **Note:** This parameter is mandatory (except when the `isBase` parameter is configured to

Parameter	Type	Description
		`true`).
`enableSpeechInput`	Boolean	Enables the activation of the STT service after the bot message playback is finished (typically, this is the trigger for activating STT). ■ `true` (Default) ■ `false` For example, if the bot wants to send two messages to the user (with some time gap between them), and therefore, doesn't want to allow speech input after the first message playback, it can disable speech input for that message using the `activityParams` object.
`name`	String	Defines the unique name of the specific bot (e.g., "LondonTube") for identification between the SBC and the Voice.AI Gateway (the name is not related to the name used in the bot framework). This name should be sent as the destination user by the SBC, for a call to be routed to the bot. For example, it can be configured using the 'Modified Destination User Name' parameter in the IP-to-IP Routing table. For more information, see Configuring the SBC. **Note:** This parameter is mandatory.
`credentials`	String	Defines the access key to the bot framework. ■ `botSecret` (applicable only to Azure) ■ `conversationProfileId` (applicable only for the One-Click Dialogflow application) ■ `sendMetaDataUsername`: Username for HTTP basic authentication. This parameter is applicable only if `sendMetaDataUrl` is configured. ■ `sendMetaDataPassword`: Password for HTTP basic authentication. This parameter is applicable only if

Parameter	Type	Description
		`sendMetaDataUrl` is configured. **Note:** The string values must be provided to AudioCodes by the Customer from the bot framework service provider. For more information, see the [Voice.AI Gateway Integration Guide]().

`credentialsAzureSecrets`	Object	The parameter is used as an alternative to the `credentials` parameter (above). Each of the attributes listed under the `credentials` parameter can be used with `credentialsAzureSecrets` parameter. Instead of plain text values, the values are specified using Azure Key Vault URLs (where the credentials are securely stored).
		The following is an example for the `key` attribute:
		`{`
		`  "name": "sample-bot",`
		`  "displayName": "Sample bot",`
		`  "provider": "my-provider",`
		`  "credentialsAzureSecrets": {`
		`    "botSecret": "https://keyvault-name.vault.azure.net/secrets/MY-SECRET/VERSION"`
		`  }`
		`}`
		You need to add the plain-text credential to Azure's Key Vault. Once added, you need to generate a URL (HTTPS) for the credential, and then use this URL ('Secret Identifier' field) as the value for the `credentialsAzureSecrets` parameter. Therefore, the credential is not shown in the Voice.AI Connector configuration file; instead, only a URL to the Key Vault is shown. When the credential is required for access, the Voice.AI Connector uses this URL to access and authenticate with the Key Vault. For more information on Azure's Key Vault feature, go to https://docs.microsoft.com/en-us/azure/key-

Parameter	Type	Description

		vault/general/secure-your-key-vault.
		Note:
		- Azure's Key Vault can be used to store secrets for any bot provider (e.g., Google or Yandex).
		- You can mix `credentials` and `credentialsAzureSecrets`, as long as you don't have the same key in both parameters. For example:
		{
		"name": "my_google",
		"type": "google",
		"projectId": "my-project",
		"credentialsAzureSecrets": {
		"privateKey": "https://keyvault-name.vault.azure.net/secrets/MY-SECRET/VERSION"
		},
		"credentials": {
		"clientEmail": "service@my-project.iam.gserviceaccount.com"
		}
		}

`generalFailoverActivities`	String	Defines activities that are executed sequentially in case of an error after the call is connected. The error can be one of the following: - Bot framework error - STT error - TTS error - SBC error This parameter is defined with the name of the `activityLists` parameter `name` value, which defines the activities to execute. For example, if an error occurs, a prompt can be played to the user. By default, the parameter is not defined.

Parameter	Type	Description
		For more information, see Activity Lists Section.
`provider`	String	Defines a name of a provider to be used for the bot service. This name references the `name` parameter of a provider configuration (see Providers Section on page 50). This provider will also be used for TTS and STT, unless the `ttsProvider` and `sttProvider` parameters respectively are configured. **Note:** This parameter is mandatory (except when the `isBase` parameter is configured to `true`).
`sttProvider`	String	Defines a name of an STT provider to be used for the bot service. This name references the `name` parameter of a provider configuration (see Providers Section on page 50). **Note:** The parameter is used only if you want to use an STT provider that's different from the bot framework provider defined by the `provider` parameter (described in this table).
`ttsProvider`	String	Defines a name of a TTS provider to be used for the bot service. This name references the `name` parameter of a provider configuration (see Providers Section on page 50). **Note:** The parameter is used only if you want to use a TTS provider that's different from the bot framework provider defined by the `provider` parameter (described in this table).
`directSTT`	Boolean	Enables the use of the bot framework's STT service instead of a standalone one. ■ `true` ■ `false` (Default) **Note:** The parameter is applicable only to Google bot framework.
`directTTS`	Boole	Enables the use of the bot framework's TTS

Parameter	Type	Description
	an	service instead of a standalone one. - `true` - `false` (Default) **Note:** The parameter is applicable only to Google bot framework.
`providerBotName`	String	Defines the bot name as defined in the bot framework provider. **Note:** The parameter is applicable only to AWS or AudioCodes bot API.
`botURL`	String	- AudioCodes Bot API: Defines the URL for sending the `start` conversation HTTP request. Note that the URL should include the path "StartConversation" (if such a path is needed). For example, `"https://localhost:8083/botURL"`. - Azure bot: Overwrites the default URL (https://directline.botframework.com/v3/directline/) that is used to create a conversation with Direct Line API.
`initialMessageToBot`	String	Defines a textual message that is sent to the bot when conversation starts.
`initialActivityToBot`	JSON	Defines a JSON object that is sent to the bot when conversation starts. This parameter overrides the default initial activity (if defined for the specific bot framework). To prevent the sending of an initial activity, configure the parameter to an empty object `"{}"`.
`welcome`	String	Defines a text to be played to the user when conversation starts.
`dialogflowWelcomeEvent`	String	Defines the name of the event to send to Dialogflow when the conversation starts. The default is "WELCOME". **Note:**

Parameter	Type	Description
		- The parameter is applicable only to the Google bot framework. - The parameter is only applicable if `initialActivityToBot` is not configured.
`controlScripts`	String	Defines the name of a control script that contains a customized hook (e.g., Control.manipulation.messageFromBot). This name references the control script in the `controlScript` section. For more information on control scripts, see controlScripts Sectioncontact AudioCodes.
`sendMetaDataUrl`	String	Defines the URL (of the HTTP server) to where the metadata is sent. If the parameter is not configured, the `sendMetaData` activity is handled by sending SIP INFO messages. **Note:** The parameter can be dynamically modified using placeholders (see Dynamically Setting Parameters on Conversation Start).
`sendMetaDataTimeoutMs`	Number	Defines the timeout (in milliseconds) for the HTTP request. The valid value is 10-60000. The default is 10,000. **Note:** The parameter is applicable only if `sendMetaDataUrl` is configured.
`callRecordingServer`	String	Defines the Session Recording Server (SRS) for SIPRec as an IP Group name (as configured on the SBC) to record the call. The SRS is used when the bot starts call recording using the `startCallRecording` action. For more information, refer to the *Voice.AI Gateway Integration Guide*. **Note:** Due to security, the parameter is not controlled by the bot. Instead, it is a required configuration for enabling usage of the `startCallRecording` action.

Parameters Controlled Also by Bot

These parameters can be configured on the Voice.AI Connector, but they can also be determined and updated by the bot dynamically. The bot takes precedence (i.e., overrides Voice.AI Connector configuration). The bot can override the specific activity (activityParams) or the entire conversation (sessionParams). For more information on activity and session parameters, refer to the Voice.AI Gateway Integration Guide.

Table 4-2: Bots Section Parameter Descriptions (Also Controlled by Bot)

Parameter	Type	Description
`azureSpeechRecognitionMode`	String	Defines the Azure STT recognition mode. ■ `conversation` (default) ■ `dictation` ■ `interactive` **Note:** The parameter is applicable only to the Microsoft Azure STT service.
`bargeIn`	Boolean	Enables the Barge-In feature. ■ `true`: Enabled, When the bot is playing a response to the user (playback of bot message), the user can "barge-in" (interrupt) and start speaking. This terminates the bot response, allowing the bot to listen to the new speech input from the user (i.e., Voice.AI Gateway sends detected utterance to the bot). ■ `false`: (Default) Disabled. The Voice.AI Gateway doesn't expect speech input from the user until the bot has finished playing its response to the user. In other words, the user can't "barge-in" until the bot message response has finished playing.
`bargeInOnDTMF`	Boolean	Enables the Barge-In on DTMF feature. ■ `true`: (Default) Enabled. When the bot is playing a response to the user (playback of bot message), the user can "barge-in" (interrupt) with a DTMF digit. This terminates the bot response, allowing the bot to listen to and process the digits sent from the user.

Parameter	Type	Description
		■ `false`: Disabled. The Voice.AI Connector doesn't expect DTMF input from the user until the bot has finished playing its response to the user. In other words, the user can't "barge-in" until the bot message response has finished playing. **Note:** ■ If you enable this feature (i.e., `bargeInOnDTMF` configured to `true`), you also need to enable the sending of DTMF digits (see the `sendDTMF` parameter). ■ Currently, this parameter is not supported when speech-to-text is performed by the bot framework.
`bargeInMinWordCount`	Number	Defines the minimum number of words that the user must say for the Voice.AI Gateway to consider it a barge-in. For example, if configured to 4 and the user only says 3 words during the bot's playback response, no barge-in occurs. The valid range is 1 to 5. The default is 1.
`botFailOnErrors`	Boolean	Defines what happens when the Azure bot error "retry" occurs. ■ `true:` The error is printed to the log and the call is disconnected. ■ `false:` (Default) The error is printed to the log, but the call is not disconnected.
`botNoInputGiveUpTimeoutMS`	Number	Defines the maximum time that the Voice.AI Connector waits for a response from the bot. If no response is received when the timeout expires, the Voice.AI Connector disconnects the call with the SBC. The default is 0 (i.e., feature disabled). If the call is disconnected, the SIP BYE message sent by the SBC to the user indicates this failure, by prefixing the value in the Reason header with "Bot Err:".

Parameter	Type	Description
		Note: In this scenario (disconnects), you can also configure the Voice.AI Connector to perform specific activities, for example, playing a prompt to the user or transferring the call (see the `generalFailoverActivities` parameter).
`botNoInputTimeoutMS`	Number	Defines the maximum time (in milliseconds) that the Voice.AI Connector waits for input from the bot framework. If no input is received from the bot when this timeout expires, you can configure the Voice.AI Connector to play a textual (see the `botNoInputSpeech` parameter) or an audio (see the botNoInputUrl parameter) prompt to the user. The default is 0 (i.e., feature disabled).
`botNoInputRetries`	Number	Defines the maximum number of allowed timeouts (configured by the `botNoInputTimeoutMS` parameter) for no bot input. If you have configured a prompt to play (see the `botNoInputSpeech` or `botNoInputUrl` parameter), the prompt is played to the user each time the timeout expires. The default is 0 (i.e., only one timeout – no retries). For more information on the no bot input feature, see the `botNoInputTimeoutMS` parameter. **Note:** If you have configured a prompt to play upon timeout expiry, the timer is triggered only after playing the prompt to the user.
`botNoInputSpeech`	String	Defines the textual prompt to play to the user when no input has been received from the bot framework when the timeout expires (configured by `botNoInputTimeoutMS`). The prompt can be configured in plain text or in Speech Synthesis Markup Language (SSML) format:

Parameter	Type	Description

		- **Plain-text example:**
		{
		"name": "LondonTube",
		"provider": "my_azure",
		"displayName": "London Tube",
		"botNoInputTimeoutMS": 5000,
		"botNoInputSpeech": "Please wait for bot input"
		}
		- **SSML example:**
		{
		"name": "LondonTube",
		"provider": "my_azure",
		"displayName": "London Tube",
		"botNoInputTimeoutMS": 5000,
		"botNoInputSpeech": <speak>"This is <say-as interpret-as="characters">SSML</say-as>"</speak>
		}
		By default, the parameter is not configured.
		Note:
		- For more information on the no bot input feature, see the `botNoInputTimeoutMS` parameter.
		- If you have also configured to play an audio prompt (see the `botNoInputUrl` parameter), the `botNoInputSpeech` takes precedence.
		- This feature requires a text-to-speech provider. It will not work when the speech is synthesized by the bot framework.
		- The supported SSML elements depend on the text-to-speech provider:
		✓ **Google:** https://cloud.google.com/text-

Parameter	Type	Description
		to-speech/docs/ssml ✔ **Azure:** https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/speech-synthesis-markup#supported-ssml-elements ✔ **AWS:** https://docs.aws.amazon.com/polly/latest/dg/supportedtags.html
`botNoInputUrl`	String	Defines the URL from where the audio prompt is played to the user when no input has been received from the bot when the timeout expires (configured by `botNoInputTimeoutMS`). By default, the parameter is not configured. For more information on the no bot input feature, see the `botNoInputTimeoutMS`. **Note:** If you have also configured to play a textual prompt (see the `botNoInputSpeech` parameter), the `botNoInputSpeech` takes precedence.
`userNoInputTimeoutMS`	Number	Defines the maximum time (in milliseconds) that the Voice.AI Connector waits for input from the user. If no input is received when this timeout expires, you can configure the Voice.AI Connector to play a textual (see the `userNoInputSpeech` parameter) or an audio (see the userNoInputUrl parameter) prompt to ask the user to say something. If there is still no input from the user, you can configure the Voice.AI Connector to prompt the user again. The number of times to prompt is configured by the `userNoInputRetries` parameter. If the `userNoInputSendEvent` parameter is configured to `true` and the timeout expires, the Voice.AI Connector sends an event to the bot, indicating how many times the timer has expired. The default is 0 (i.e., feature disabled). **Note:**

Parameter	Type	Description
		- DTMF (any input) is considered as user input (in addition to user speech) if the `sendDTMF` parameter is configured to `true`. - If you have configured a prompt to play when the timeout expires, the timer is triggered only after playing the prompt to the user.
`userNoInputRetries`	Number	Defines the maximum number of allowed timeouts (configured by the `userNoInputTimeoutMS` parameter) for no user input. If you have configured a prompt to play (see the `userNoInputSpeech` or userNoInputUrl parameter), the prompt is payed each time the timeout expires. The default is 0 (i.e., only one timeout). For more information on the no user input feature, see the `userNoInputTimeoutMS` parameter. **Note:** If you have configured a prompt to play upon timeout expiry, the timer is triggered only after playing the prompt to the user.

`userNoInputSendEvent`	Boolean	Enables the Voice.AI Connector to send an event message to the bot if there is no user input for the duration configured by the `userNoInputTimeoutMS` parameter, indicating how many times the timer has expired ('value' field):
		`{`
		`"type": "event",`
		`"name": "noUserInput",`
		`"value": 1`
		`}`
		- `true`: Enabled.
		- `false`: (Default) Disabled.
		Note: The feature is applicable only to Azure, Google, and AudioCodes API (`ac-api`).

Parameter	Type	Description

`userNoInputSpeech`	String	Defines the textual prompt to play to the user when no input has been received from the user when the timeout expires (configured by `userNoInputTimeoutMS`).
		The prompt can be configured in plain text or in Speech Synthesis Markup Language (SSML) format:
		By default, the parameter is not configured.
		- **Plain-text example:**
		{
		"name": "LondonTube",
		"provider": "my_azure",
		"displayName": "London Tube",
		"userNoInputTimeoutMS": 5000,
		"userNoInputSpeech": "Hi there. Please say something"
		}
		- **SSML example:**
		{
		"name": "LondonTube",
		"provider": "my_azure",
		"displayName": "London Tube",
		"userNoInputTimeoutMS": 5000,
		"userNoInputSpeech": <speak>"This is <say-as interpret-as="characters">SSML</say-as>"</speak>
		}
		For more information on the no user input feature, see the `userNoInputTimeoutMS`.
		Note:
		- If you have also configured to play an audio prompt (see the `userNoInputUrl` parameter), the `userNoInputSpeech` takes precedence.

Parameter	Type	Description
		- This feature requires a text-to-speech provider. It will not work when the speech is synthesized by the bot framework. - The supported SSML elements depend on the text-to-speech provider: ✓ **Google:** https://cloud.google.com/text-to-speech/docs/ssml ✓ **Azure:** https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/speech-synthesis-markup#supported-ssml-elements ✓ **AWS:** https://docs.aws.amazon.com/polly/latest/dg/supportedtags.html
`userNoInputUrl`	String	Defines the URL from where the audio prompt is played to the user when no input has been received from the user when the timeout expires (configured by `userNoInputTimeoutMS`). By default, the parameter is not configured. For more information on the no user input feature, see the `userNoInputTimeoutMS.` **Note:** If you have also configured to play a textual prompt (see the `userNoInputSpeech` parameter), the `userNoInputSpeech` takes precedence.
`continuousASR`	Boolean	Enables the Continuous ASR feature. Continuous ASR enables the Voice.AI Gateway to concatenate multiple STT recognitions of the user and then send them as a single textual message to the bot. - `true`: Enabled - `false`: (Default) Disabled For an overview of the Continuous ASR feature, refer to the Voice.AI Gateway Product Description.
`continuousASRDigits`	String	This parameter is applicable when the Continuous ASR feature is enabled.

Parameter	Type	Description
		Defines a special DTMF key, which if pressed, causes the Voice.AI Gateway to immediately send the accumulated recognitions of the user to the bot. For example, if configured to "#" and the user presses the pound key (#) on the phone's keypad, the device concatenates the accumulated recognitions and then sends them as one single textual message to the bot. The default is "#". **Note:** Using this feature incurs an additional delay from the user's perspective because the speech is not sent immediately to the bot after it has been recognized. To overcome this delay, configure the parameter to a value that is appropriate to your environment.
`continuousASRTimeoutInMS`	Number	This parameter is applicable when the Continuous ASR feature is enabled. Defines the automatic speech recognition (ASR) timeout (in milliseconds). When the device detects silence from the user for a duration configured by this parameter, it concatenates all the accumulated STT recognitions and sends them as one single textual message to the bot. The valid value is 2,500 (i.e., 2.5 seconds) to 60,000 (i.e., 1 minute). The default is 3,000.
`disableTtsCache`	Boolean	Defines caching of TTS (audio) results from the bot. Therefore, if the Voice.AI Connector needs to send a request for TTS to a TTS provider and this text has been requested before, it retrieves the result from its cache instead of requesting it again from the TTS provider. ■ `true`: TTS caching is disabled. ■ `false`: (Default) TTS caching is enabled. **Note:** This parameter is not applicable when speech-to-text is performed by the bot framework.
`googleInteractionTy`	Stri	Defines the Google STT interaction type. For more

Parameter	Type	Description
`pe`	ng	information, see https://cloud.google.com/speech-to-text/docs/reference/rest/v1p1beta1/RecognitionConfig#InteractionType.
`handoverReason`	String	Defines the textual reason when the call is transferred to another party (e.g., another bot or a human agent). By default, the parameter is not defined.

`hangupReason`	String	Conveys a textual reason for hanging up (disconnecting call). This reason appears in the CDR of the call.
		Example message:
		`{`
		`"type": "event",`
		`"name": "hangup",`
		`"activityParams": {`
		`"hangupReason": "conversationCompleted"`
		`}`
		`}`

`language`	String	Defines the language (e.g., "en-ZA" for South African English) of the bot conversation and is used for TTS and STT functionality. The value is obtained from the service provider. ■ **STT:** ✔ Azure: The parameter is configured with the value from the 'Locale' column in Azure's Speech-Text table (e.g., "en-GB"). ✔ Google: The parameter is configured with the value from the 'languageCode' (BCP-47) column in Google's Cloud Speech-to-Text table (e.g., "nl-NL"). ■ **TTS:** ✔ Azure: The parameter is configured with the value from the 'Locale' column in

Parameter	Type	Description
		Azure's Text-to-Speech table (e.g., "it-IT"). ✔ Google: The parameter is configured with the value from the 'Language code' column in Google's Cloud Text-to-Speech table (e.g., "en-US"). ✔ AWS: The parameter is configured with the value from the 'Language' column in Amazon's Polly TTS table (e.g., "de-DE"). **Note:** This string is obtained from the TTS or STT service provider by the Customer and must be provided to AudioCodes. For more information, see the Voice.AI Gateway Integration Guide.
`playUrlAltText`	String	Defines the text to display in the transcript page of the user interface while the audio is played.
`playUrlCaching`	Boolean	Enables caching of the audio in the TTS cache: ■ `true`: Enables caching ■ `false`: (Default) Disables caching
`playUrlMediaFormat`	String	Defines the format of the audio: ■ `wav/lpcm16` (default) ■ `raw/lpcm16`
`playUrlUrl`	String	Defines the HTTP-based server by URL where the audio file to be played is located. This allows the play of pre-recorded prompts (audio file) to the user from a remote third-party server.
`resumeRecognitionTimeoutMS`	Number	When Barge-In is disabled, speech input is not expected before the bot's response has finished playback. If no reply from the bot arrives within this configured timeout (in milliseconds), the Voice.AI Gateway expects speech input from the user and STT recognition is re-activated. The valid value is 0 (i.e., no automatic resumption of recognition) to 600,000 (i.e., 10 minutes). The default is 10,000.
`sendDTMF`	Bool	Enables the sending of DTMF events to the bot.

Parameter	Type	Description
	ean	- `true`: Enabled - `false`: (Default) Disabled **Note:** For configuring the DTMF collection and sending method, see the `dtmfCollect` parameter.
`dtmfCollect`	Boolean	Defines the DTMF digit collection and sending method. - `true`: Enabled. The Voice.AI Gateway first collects all the DTMF digits entered by the user, and only then sends them all together to the bot. - `false`: (Default) Disabled. As the Voice.AI Gateway receives a DTMF digit entered by the user, it sends that single digit to the bot. In other words, it sends each DTMF digit one at a time to the bot. **Note:** - When enabled, you can configure additional settings using the following parameters: `dtmfCollectInterDigitTimeoutMS`, `dtmfCollectMaxDigits`, and `dtmfCollectSubmitDigit`. - If the `sendDTMF` parameter is configured to `false` (default), incoming DTMF digits are ignored by the Voice.AI Gateway even if the `dtmfCollect` parameter is configured to `true`. - DTMF collection is not applicable to Dialogflow CX bots, as it is performed by the bot framework.
`dtmfCollectInterDigitTimeoutMS`	Number	Defines the timeout (in milliseconds) that the Voice.AI Gateway waits for the user to press another digit before it sends all the digits to the bot. If the timeout expires since the last digit entered by the user, the Voice.AI Gateway sends all the collected digits to the bot (as a DTMF

Parameter	Type	Description
		message), without waiting for the maximum number of expected digits or for the "submit" digit. The timeout is triggered after the user enters the first DTMF digit and is reset after each digit. The valid value range is 0 to unlimited. The default is 2000. **Note:** - The parameter is applicable only when the `dtmfCollect` parameter is configured to `true`. - Once the Voice.AI Gateway sends all the DTMF digits to the bot, any additional DTMF digits entered by the user is ignored by the Voice.AI Gateway until the bot responds to the DTMF event message.
`dtmfCollectMaxDigits`	Number	Defines the maximum number of DTMF digits that the Voice.AI Gateway expects to receive from the user. Once the Voice.AI Gateway receives and collects this number of digits entered by the user, it immediately sends all the digits to the bot (as a DTMF message), without waiting for the timeout to expire or for the "submit" digit. The valid value range is 0 (disabled) to unlimited. The default is 5. If configured to 0, the DTMF collection and sending method is according to `dtmfCollectInterDigitTimeoutMS` or `dtmfCollectSubmitDigit`. **Note:** - The parameter is applicable only when the `dtmfCollect` parameter is configured to `true`. - Once the Voice.AI Gateway sends all the DTMF digits to the bot, any additional DTMF digits entered by the user is ignored by the Voice.AI Gateway until the bot responds to the DTMF event message.
`dtmfCollectSubmitDi`	Stri	Defines a special DTMF "submit" digit that when

Parameter	Type	Description
`git`	ng	received from the user, the Voice.AI Gateway immediately sends all the collected digits to the bot (as a DTMF message), without waiting for the timeout to expire or for the maximum number of expected digits. The valid value is any symbol on a phone keypad. The default is # (pound key). If you want to disable this parameter, configure it to "" (empty string). **Note:** ■ The parameter is applicable only when the `dtmfCollect` parameter is configured to `true`. ■ The Voice.AI Gateway doesn't include this "submit" digit in the DTMF event message sent to the bot. ■ Once the Voice.AI Gateway sends all the DTMF digits to the bot, any additional DTMF digits entered by the user is ignored by the Voice.AI Gateway until the bot responds to the DTMF event message.
`sttContextId`	String	■ **Azure speech-to-text engine:** This parameter controls Azure's Custom Speech model. The parameter can be set to the endpoint ID that is used when accessing the STT engine. For more information on how to obtain the endpoint ID, go to https://docs.microsoft.com/en-us/azure/cognitive-services/speech-service/how-to-custom-speech-deploy-model. ■ **AudioCodes DNN speech-to-text engine:** This parameter controls the context. **Note:** ■ The parameter can be used by all bot providers, as long as the STT engine is Azure or AudioCodes DNN. ■ For Azure STT, the Custom Speech model must be deployed on the same subscription

Parameter	Type	Description
		used for the Azure STT engine. ■ When using other STT engines, the parameter has no affect.
`sttContextPhrases`	Array of Strings	When using Google's Cloud STT engine, this parameter controls Speech Context phrases. The parameter can list phrases or words that is passed to the STT engine as "hints" for improving the accuracy of speech recognitions. For more information on speech context (speech adaptation) as well details regarding tokens (class tokens) that can be used in phrases, go to https://cloud.google.com/speech-to-text/docs/speech-adaptation. For example, whenever a speaker says "weather" frequently, you want the STT engine to transcribe it as "weather" and not "whether". To do this, the parameter can be used to create a context for this word (and other similar phrases associated with weather): `"sttContextPhrases": ["weather"]` **Note:** ■ The parameter is only for backward compatibility and will be deprecated in the next applicable release. Please use the `sttSpeechContexts` parameter instead. ■ The parameter can be used by all bot providers when the STT engine is Google. ■ When using other STT engines, the parameter has no affect.
`sttContextBoost`	Number	Defines the boost number for context recognition of the speech context phrase configured by `sttContextPhrases`. Speech-adaptation boost allows you to increase the recognition model bias by assigning more weight to some phrases than others. For example, when users say "weather" or "whether", you may want the STT to

Parameter	Type	Description
		recognize the word as weather. For more information, see https://cloud.google.com/speech-to-text/docs/context-strength. **Note:** - The parameter is only for backward compatibility and will be deprecated in the next applicable release. Please use the `sttSpeechContexts` parameter instead. - The parameter can be used by all bot providers when the STT engine is Google. - When using other STT engines, the parameter has no affect.
`sttSpeechContexts`	Array of Strings	When using Google's Cloud STT engine, this parameter controls Speech Context phrases. The parameter can list phrases or words that is passed to the STT engine as "hints" for improving the accuracy of speech recognitions. For example, whenever a speaker says "weather" frequently, you want the STT engine to transcribe it as "weather" and not "whether". To do this, the parameter can be used to create a context for this word (and other similar phrases associated with weather). You can also use the parameter to define the boost number (0 to 20, where 20 is the highest) for context recognition of the specified speech context phrase. Speech-adaptation boost allows you to increase the recognition model bias by assigning more weight to some phrases than others. For example, when users say "weather" or "whether", you may want the STT to recognize the word as "weather". For more information, see https://cloud.google.com/speech-to-text/docs/context-strength. You can also use Google's class tokens to represent common concepts that occur in natural language, such as monetary units and calendar

Parameter	Type	Description

		dates. A class allows you to improve transcription accuracy for large groups of words that map to a common concept, but that don't always include identical words or phrases. For example, the audio data may include recordings of people saying their street address. One may say "my house is 123 Main Street, the fourth house on the left." In this case, you want Speech-to-Text to recognize the first sequence of numerals ("123") as an address rather than as an ordinal ("one-hundred twenty-third"). However, not all people live at "123 Main Street" and it's impractical to list every possible street address in a SpeechContext object. Instead, you can use a class token in the phrases field of the SpeechContext object to indicate that a street number should be recognized no matter what the number actually is.
		For example:
		"sttSpeechContexts": [{
		  "phrases”: ["weather"],
		  "boost": 18
		  },
		 {
		  "phrases”: ["whether"],
		  "boost": 2
		  },
		 {
		  "phrases”: ["fair"]
		  },
		  {
		  “phrases”: [“$ADDRESSNUM”]
		  }
		]
		Note:
		- The parameter can be used by all bot providers when the STT engine is Google.
		- When using other STT engines, the parameter has no affect.

Parameter	Type	Description
		- For more information on speech context (speech adaptation) as well details regarding tokens (class tokens) that can be used in phrases, go to https://cloud.google.com/speech-to-text/docs/speech-adaptation.
`sttDisablePunctuation`	Boolean	Prevents the STT response from the bot to include punctuation marks. - `true`: Enabled. Punctuation is excluded. - `false`: (Default) Disabled. Punctuation is included. **Note:** This requires support from the STT engine.
`sttEndpointID`	String	This parameter has been deprecated in Version 2.2 and replaced by the `sttContextId` parameter.
`azureEnableAudioLogging`	Boolean	Enables recording and logging of audio from the user (endpoint) that the Voice.AI Gateway sends to the STT engine. The recording is done by the STT engine and stored on the STT engine. - `true`: Instructs the STT engine to enable audio logging. - `false`: Instructs the STT engine to disable audio logging. When the parameter is not defined (default), audio logging is according to the STT engine. **Note:** The parameter and audio logging is applicable only when using the Azure STT.
`targetParticipant`	String	Defines the participant on which to apply the events `startRecognition` and `stopRecognition` for starting and stopping (respectively) speech recognition by the STT engine. **Note:** The parameter is applicable only to Agent Assist calls.

Parameter	Type	Description
`transferReferredByURL`	String	Defines the party (URL) who initiated the referral. If this parameter exists, the SBC adds a SIP Referred-By header to the outgoing INVITE/REFER message (according to the 'Remote REFER Mode' parameter). If the SBC handles locally (termination), the SBC adds it to a new outgoing INVITE. If not handled locally (regular), the SBC adds it to the forwarded REFER message.
`transferSipHeaders`	Array of Objects	Array of objects listing SIP headers that should be sent to the transferee. Each object comprises a `name` and a `value` attribute.
`transferTarget`	String	Defines the URI to where the call must be transferred. Typically, the URI is a "tel" or "sip" URI.
`voiceName`	String	Defines the voice name for the TTS service. ■ Azure: The parameter is configured with the value from the 'Short voice name' column in Azure's Text-to-Speech table (e.g., "it-IT-ElsaNeural"). ■ Google: The parameter is configured with the value from the 'Voice name' column in Google's Cloud Text-to-Speech table (e.g., "en-US-Wavenet-A"). ■ AWS: The parameter is configured with the value from the 'Name/ID' column in Amazon's Polly TTS table (e.g., "Hans"). ■ Almagu: The parameter is configured with the value from the 'Voice' column in Almagu's TTS table (e.g., "Osnat"). **Note:** This string is obtained from the TTS service provider by the Customer and must be provided to AudioCodes. For more information, see the Voice.AI Gateway Integration Guide.
`callRecordingId`	String	Defines the recording ID session, which is forwarded by the SBC to the SRS in the XML body

Parameter	Type	Description

		(attribute "ac:call-recording-id") of the SIP INVITE request for SIPRec. The Administrator can later retrieve the bot's recordings from the SRS, by using this ID.
		The parameter is used by the feature where the bot starts and stops voice recording of the conversation. For more information, refer to the *Voice.AI Gateway Integration Guide*.
		Below is an example of an XML body with the ID shown in the XML attribute "ac:call-recording-id":
		`<?xml version="1.0" encoding="UTF-8"?>`
		`<recording xmlns="urn:ietf:params:xml:ns:recording" xmlns:ac="http://AudioCodes">`
		`<datamode>complete</datamode>`
		`<group id="00000000-0000-0072-57a7-310000000072">`
		`<associate-time>2018-01-01T02:22:29</associate-time>`
		`</group>`
		`<session id="0000-0000-0000-0000-b44497aaf9597f7f">`
		`<group-ref>00000000-0000-0072-57a7-310000000072</group-ref>`
		`<associate-time>2018-01-01T02:22:29</associate-time>`
		`<`**`ac:call-recording-id`**`>KUYdgtofldi76YpkP09J4J-a:1</ac:call-recording-id>`
		`</session>`
		`<participant id="+123456789" session="0000-0000-0000-0000-b44497aaf9597f7f">`
		`....`
		Note:
		■ The parameter is applicable only to the `startCallRecording` event.
		■ The XML attribute name "ac:call-recording-id"

Parameter	Type	Description
		can be changed using the SBC's Message Manipulation feature.
`callRecordingDestUsername`	String	Defines the username that is used in the SIP Request-URI and To headers of the INVITE request for SIPRec. The parameter is used by the feature where the bot starts and stops voice recording of the conversation. For more information, refer to the *Voice.AI Gateway Integration Guide*. **Note:** The parameter is applicable only to the `startCallRecording` event.

5 Providers Section

The `providers` section configures the details of each external bot service (bot framework, STT engine, or TTS engine) that is used. Some providers support multiple services; others are limited to a specific service only.

Each provider is identified by the `name` parameter, which must be unique in the Voice.AI Gateway configuration. This name is used to reference the provider in the bot's configuration.

As each bot in the `bots` section references one or more providers, the `providers` section must contain the configuration for all the referenced providers.

> ⚠ These parameters are controlled only by the administrator of the Voice.AI Connector, using the REST API or configuration file.

For example, the following configures two bot providers, AWS and Azure:

```
"providers": [
 {
  "name": "my_aws",
  "type": "aws",
  "region": "us-west-2",
  "credentials": {
   "accessKey": "AKIAIHHFDNMRA",
   "secretKey": "fqDC7STjK1elH1g"
  }
 },
 {
  "name": "my_azure",
  "type": "azure",
  "region": "westeurope",
  "credentials": {
   "key": "ab6612c69ec1df5d67"
  }
 }
]
```

Figure 5-1: Providers Section Parameter Descriptions

Parameter	Type	Description
`botAllowSelfSignedCert`	Boolean	Enables only connections to the bot server that use a self-signed certificate; otherwise, connection attempts are rejected. ■ `true`

Parameter	Type	Description
		- `false` (Default)
`oauthTokenUrl`	String	Defines the URL of the OAuth 2.0 authorization server that provides the access token for OAuth authorization when the AC Bot API is used. Configuring the parameter implies that OAuth 2.0 authentication is enabled. For more information, refer to the *Voice.AI Gateway API Reference Guide*. **Note:** - The value must be provided to AudioCodes by the Customer. - If the parameter is configured, you also need to configure the `oauthClientId` and `oauthClientSecret` parameters.
`oauthScope`	String	Defines the OAuth scope (permissions) that is sent in the request to the OAuth 2.0 authorization server that provides the access token for OAuth authorization when the AC Bot API is used. For more information, refer to the *Voice.AI Gateway API Reference Guide*. The parameter can be configured with multiple scopes. **Note:** - The value is optional and is provided to AudioCodes by the Customer. - If the parameter is configured, you also need to configure the `oauthClientId` and `oauthClientSecret` parameters.
`credentials`	Object	Contains the credentials to access the bot framework, STT, or TTS. The parameter has the following optional fields: - `accessKey` (applicable only to AWS) - `secretKey` (applicable only to AWS) - `key` (applicable only to Azure or Yandex speech services).

Parameter	Type	Description
		■ `privateKey` (applicable only to Google Dialogflow) ■ `clientEmail` (applicable only to Google Dialogflow) ■ `token` (applicable only to Membit or AudioCodes Bot API) ■ `overrideAccessToken` (used only for internal AudioCodes testing) ■ `oauthClientId`: The Client ID provided by the API for OAuth 2.0 authorization. This is mandatory if the `oauthTokenUrl` parameter is configured. This parameter is applicable only to the AC Bot API. For more information on OAuth 2.0 authorization, refer to the *Voice.AI Gateway API Reference Guide*. ■ `oauthClientSecret`: The Client secret provided by the API for OAuth 2.0 authorization. This is mandatory if the `oauthTokenUrl` parameter is configured. This parameter is applicable only to the AC Bot API. For more information on OAuth 2.0 authorization, refer to the *Voice.AI Gateway API Reference Guide*. **Note:** ■ The value must be provided to AudioCodes by the Customer from the service providers. For more information, see the Voice.AI Gateway Integration Guide. ■ Each of the above secret types can be securely stored in **Azure's Key Vault** (preventing attackers from obtaining these keys). If this is the case, instead of using the `credentials` parameter, use the `credentialsAzureSecrets` parameter, with the same attribute names as listed above.

Parameter	Type	Description

`credentialsAzureSecrets`	Object	The parameter is used as an alternative to the `credentials` parameter (above). Each of the attributes listed under the `credentials` parameter can be used with `credentialsAzureSecrets` parameter. Instead of plain text values, the values are specified using Azure Key Vault URLs (where the credentials are securely stored).
		The following is an example for the `key` attribute:
		`{`
		`  "name": "my_azure",`
		`  "type": "azure",`
		`    "region": "westeurope",`
		`    "credentialsAzureSecrets": {`
		`       "key": "https://keyvault-name.vault.azure.net/secrets/MY-SECRET/VERSION"`
		`     }`
		`  }`
		You need to add the plain-text credential to Azure's Key Vault. Once added, you need to generate a URL (HTTPS) for the credential, and then use this URL ('Secret Identifier' field) as the value for the `credentialsAzureSecrets` parameter. Therefore, the credential is not shown in the Voice.AI Connector configuration file; instead, only a URL to the Key Vault is shown. When the credential is required for access, the Voice.AI Connector uses this URL to access and authenticate with the Key Vault. For more information on Azure's Key Vault feature, go to https://docs.microsoft.com/en-us/azure/key-vault/general/secure-your-key-vault.
		Note:
		- Azure's Key Vault can be used to store secrets for any bot provider (e.g., Google or Yandex).
		- You can mix `credentials` and

Parameter	Type	Description

		`credentialsAzureSecrets`, as long as you don't have the same key in both parameters. For example:
		{
		"name": "my_google",
		"type": "google",
		"projectId": "my-project",
		"credentialsAzureSecrets": {
		"privateKey": "https://keyvault-name.vault.azure.net/secrets/MY-SECRET/VERSION"
		},
		"credentials": {
		"clientEmail": "service@my-project.iam.gserviceaccount.com"
		}
		}

`name`	String	Defines a unique name identifier of the specific provider. This name is used by the `provider`, `ttsProvider` or `sttProvider` bot parameters for associating the bot with this provider (see Bots Section on page 21). **Note:** This parameter is mandatory.
`type`	String	Defines the name of the bot framework provider. ■ `aws` ■ `azure` ■ `google` ■ `ac-api` (applicable only to AudioCodes API) ■ `yandex` ■ `membit` ■ `almagu` ■ `azure-mock` (used only for AudioCodes internal testing)

Parameter	Type	Description
		Note: - This parameter is mandatory. - `ac-bot-api` has been deprecated and replaced by `ac-api`.
`region`	String	Defines the AWS Region where the bot framework is geographical located (e.g., "us-west-2"). **Note:** - The parameter is applicable only to AWS and Azure. - This string is obtained from the bot framework provider by the Customer and must be provided to AudioCodes. For more information, see the Voice.AI Gateway Integration Guide.
`projectId`	String	Defines the Project ID of the bot. This is needed for connecting to Google Dialogflow. **Note:** - The parameter is applicable only to Google Dialogflow. - This string is obtained from the bot framework provider by the Customer and must be provided to AudioCodes. For more information, see the Voice.AI Gateway Integration Guide.
`folderId`	String	Defines the Folder ID for Yandex bot. **Note:** - The parameter is applicable only to Yandex. - This string is obtained from the bot framework provider by the Customer and must be provided to AudioCodes. For more information, see the Voice.AI Gateway Integration Guide.
`sttHost`	String	Defines the IP address and port of the STT service provider. This overrides the STT

Parameter	Type	Description
		hostname when, for example, an HTTP proxy is used between the Voice.AI Connector and STT provider. This is used only for AudioCodes DNN server.
`ttsHost`	String	Defines the IP address and port of the TTS service provider. This overrides the TTS hostname when, for example, an HTTP proxy is used between the Voice.AI Connector and TTS provider.

`ttsUrl`	String	Defines the URL of the TTS service provider. This overrides the TTS URL when, for example, an HTTP proxy is used between the Voice.AI Connector and TTS provider.
		The following example uses AudioCodes API (`ac-api`) for integration with a third-party TTS vendor:
		{
		"name": "my_ac_api",
		 "type": "ac-api",
		 "ttsUrl": "https://localhost:8043/cognitiveservices/v1",
		 "credentials": {
		 "key": "AQVNy2mt2NAD8WZm91IEwtGa1g8o3fepJIAZQuDz"
		 }
		}
		Note: This parameter has precedence over the `ttsHost` parameter.

`sttUrl`	String	Defines the URL of the STT service provider. This overrides the STT URL when, for example, an HTTP proxy is used between the Voice.AI Connector and STT provider. The following example uses AudioCodes API (`ac-api`) for integration with a third-party STT vendor:

Parameter	Type	Description

		{
		"name": "my_ac_api",
		"type": "ac-api",
		"sttUrl":
		"https://localhost:8043/mySttServiceUrl",
		"credentials": {
		"key":
		"AQVNy2mt2NAD8WZm91IEwtGa1g8o3fepJIAZQuDz"
		}
		}
		Note: This parameter has precedence over the `sttHost` parameter.

`ttsConnectionTimeoutMS`	Integer	Defines the maximum time (in milliseconds) that the Voice.AI Connector waits for a response from the TTS provider. The valid value is 10 to 600,000. The default is 10,000. If no response is received when this timeout expires, the Voice.AI Connector tries one more time to communicate with the TTS. If still no response, the Voice.AI Connector disconnects the call with the SBC. If the call is disconnected, the SIP BYE message sent by the SBC to the user indicates this failure, by prefixing the value in the Reason header with "TTS Err:". **Note:** In this scenario (disconnect), you can also configure the Voice.AI Connector to perform specific activities, for example, playing a prompt to the user or transferring the call (see the `generalFailoverActivities` parameter).
`sttConnectionTimeoutMS`	Integer	Defines the maximum time (in milliseconds) that the Voice.AI Connector waits for a response from the STT provider. The valid value is 10 to 600,000. The default is 10,000.

Parameter	Type	Description
		If no response is received when this timeout expires, the Voice.AI Connector tries one more time to communicate with the STT. If still no response, the Voice.AI Connector disconnects the call with the SBC. If the call is disconnected, the SIP BYE message sent by the SBC to the user indicates this failure, by prefixing the value in the Reason header with "STT Err:". **Note:** In this scenario (disconnect), you can also configure the Voice.AI Connector to perform specific activities, for example, playing a prompt to the user or transferring the call (see the `generalFailoverActivities` parameter).
`sttOverrideConfig`	Object	Overrides parameters that are used when activating the STT request. The value is a JSON object comprising key-value pairs denoting the parameters and their desired value. For example: `{"sampleRateHertz": 16384 }` **Note:** - The parameters and values must suit the API used for performing the STT (according to the STT provider). - The parameter is currently applicable only to Google STT.
`sttOverrideStreamingConfig`	Object	Overrides streaming-configuration parameters that are used when activating the STT request. The value is a JSON object comprising key-value pairs denoting the parameters and their desired value. For example: `{"singleUtterance": false }` **Note:**

Parameter	Type	Description
		- The parameters and values must suit the API used for performing the STT (according to the STT provider). - The parameter is currently applicable only to Google STT.
`maxAzureFailureRetries`	Integer	Defines the number of times the Voice.AI Connector attempts to send a message to an Azure bot before failing due to a timeout. The valid range is 0 to 5. The default is 0. **Note:** The parameter is applicable only to Azure.
`reorderQueueSize`	Integer	Defines the maximum size of the Azure bot's ordered work queue. The valid value is 0 to 10. The default is 2. **Note:** The parameter is applicable only to Azure.
`reorderQueueTimeoutMS`	Integer	Defines the timeout for Azure bot actions. The valid value is 10 to 10,000 milliseconds. The default is 300. **Note:** The parameter is applicable only to Azure.
`msgToBotQueueingTimeoutMS`	Number	Defines the timeout (in milliseconds) that is triggered when sending DTMF digits, entered by the user, to the bot. The Voice.AI Connector sends each collected digit consecutively (in sequence) to the bot using the `DTMF` event message. The Voice.AI Connector sends the first digit immediately and then sends the next digit only when it receives a response from the bot (ACK / message posted) or this timeout expires (whichever occurs first). For example, assume that the user enters digits "158#" and the timeout is configured to 600. The Voice.AI Connector first sends a `DTMF` event message with value "1". When it receives a response from the bot or 600 msec elapses (whichever occurs first), it sends another `DTMF` event message with value "5". When it receives

Parameter	Type	Description
		a response from the bot or 600 msec elapses (whichever occurs first), it sends another `DTMF` event message with value "8", and so on until all the collected digits have been sent to the bot. The valid value range is 0 (i.e., disabled) to 10,000. The default is 800 (i.e., feature is enabled by default). **Note:** ■ The parameter is applicable only to Azure bots. ■ The parameter is applicable only if the `DtmfCollect` parameter is configured to `false` (i.e., sending of DTMF digits to the bot is one by one).

This page is intentionally left blank.

International Headquarters

1 Hayarden Street,

Airport City

Lod 7019900, Israel

Tel: +972-3-976-4000

Fax: +972-3-976-4040

AudioCodes Inc.

200 Cottontail Lane

Suite A101E

Somerset NJ 08873

Tel: +1-732-469-0880

Fax: +1-732-469-2298

Contact us: https://www.audiocodes.com/corporate/offices-worldwide

Website: https://www.audiocodes.com/

Documentation Feedback: https://online.audiocodes.com/documentation-feedback

Document #: LTRT-31020