VoiSentry

API Guide

Revision 1.27V | 2020

aculab

PROPRIETARY INFORMATION

Document Revision

Rev	Date	By	Detail
1.0d	07/03/18	rs	First release of API
1.1	12/03/18	df	Formatting
1.2	13/03/18	rs	Updates
1.3	14/03/18	rs	Corrections to recognise_verify
1.4	21/03/18	rs	Added threshold info to verify
1.5	05/04/18	rs	Added section on thresholds
1.6	09/04/18	rs	Added info on storing threshold info
1.7	15/05/18	ebj	Reformatted and updated. Audio source can be URL. Failed audio sources listed in results. No 8 kHz audio is resampled.
1.8	18/05/18	rs	Added that JSON result is UTF-8 encoded
1.9	19/06/18	df/ebj/rs	The verify service no longer accepts a threshold and does not return pass/fail. Returned confidence is now centred around 1.
1.10	24/07/18	df	Re-ordering API
1.11	10/08/18	df/ebj	Added Spanish and Italian and sensitivity
1.12	06/09/18	rs	Updated sections 3 and 4
1.13	11/12/18	rs	Added sections for Identification and Similarity methods
1.14	30/01/19	rs/ebj	Added PAD info to verify return
1.15	01/05/19	rs/ebj	Updated similarity. Added text dependent mode to verify
1.16	14/05/19	rs/ebj	Updated Identification to include threshold
1.17	17/05/19	ac/rs/ebj	Updates
1.18	20/05/19	rs	Removed references to tuples, updated examples
1.19	11/06/19	rs	File size 5 megabytes changed to thirty seconds
1.20	11/07/19	rs/ebj	Added memfree and error 507. Simplified PAD info
1.21	29/07/19	rs	Thirty second audio limit changed to 60. Added check for bit depth, minimum 16. Added PAD and sensitivity to Identify. Fixed some errors.
1.22	26/09/19	rs	API for PAD updated. Added "verified" key. "verified" is True or False
1.23	02/10/19	rs/ebj	Removed PAD info from identification
1.24	29/10/19	ac	Turn off PAD by default
1.25	30/10/19	sb/ebj	Standardised terminology and restructured
1.26	04/11/19	sb	Change Type C to Type A in PAD description

Rev	Date	By	Detail
1.27	12/11/19	rs/ebj	Fixed service name in examples
1.27V	19/02/20	ebj	Style changes implemented

CONTENTS

1 WEB SERVICES

The services exposed by this API enable Speaker Verification and Identification (SVI), Automatic Speech Recognition (ASR) and DTMF ("Touch-Tone") recognition.

1.1 SERVICES

The services below are exposed by the API. They are described in Section 8.

- **enrol** - Enrol a speaker.
- **verify** - Verify a speaker.
- **update** - Update a speaker.
- **identify** - Identify a speaker from a list of IDs.
- **reenrol** - Re-enrol a speaker.
- **recognise** - Recognise spoken words and/or DTMF key-presses.
- **recognise_verify** - Verify a speaker and recognise words / DTMF.
- **similarity** - Calculate a similarity score for two or more audio sources.
- **ping** - Ping the system, get some system information.
- **quotas** - Retrieve the quota information for a user access key.
- **query** - Query one or more speakers. Retrieve speaker statistics.
- **check** - Check whether one or more enrolment IDs exist.
- **delete** - Delete one or more speakers from the system.

1.2 RESULTS

The services return a JSON encoded results or error object. This object will take the form:

```
{
    'status' : STATUS,
    'result' : RESULT_OBJECT
}
```

`status` is a code that will be 200 if the service ran correctly. If an error occurred it will be as described in Section 1.3.

If `status` is 200, `result` will be a JSON object and will always contain:

dataset - The dataset being worked on.

tenant - The current tenant.

service - The API service that was run.

transactionid - A unique ID for this transaction.

diskalert - True if the disk usage is greater than the alert level - take action.

memfree - The amount of free memory as a percentage. If this drops below 5%, VoiSentry will stop processing verifications, identifications and enrolments.

nodestatus - 'A' active, 'B' blocked. If 'B', node may be going out of service, don't sent it more traffic.

Other relevant information is added to `result` depending on the function that is called.

If `status` is not 200, `result` will be an error string.

1.3 ERRORS

Each Web Service will raise an HTTPError if something is wrong. There are a number of codes that might be returned:

400 - Bad API request: [error message] (the service has been used incorrectly)

405 - Bad API request: method [method] not allowed.

408 - Database timeout error or Database access error.

429 - Several possible descriptions:

- System database volume too full for enrolments or updates.
- Cluster enrolment quota exceeded.
- Tenant enrolment quota exceeded.
- Accesskey enrolment quota exceeded.

435 - Has several possible descriptions:

- The audio file contains no data.
- The audio is in an unknown format.
- The sample rate is too low (below 8 kHz).
- The bit depth is too low (less than 16 and not A-Law or mu-Law).
- The audio duration is too long (more than 60 seconds).
- The audio is not single-channel (mono).

445 - Insufficient speaker observations in the audio source (implies that the audio offered to the service did not contain enough speech for VoiSentry to run its operations).

500 - An exception has occurred; a description of the problem is given.

503 - Service temporarily disabled for this key or Service temporarily disabled for this tenant.

507 - Not enough disk space or memory to fulfil the task.

2 SVI VERSIONING

Whenever a speaker model is created using the `enrol` service, the latest version of SVI will be used to process the enrolment. The SVI version is stored in the speaker model so that future verifications and updates will use the same SVI version that was used during enrolment.

If a software update installs a later version of SVI, new enrolments will use the later version but existing speaker models will continue to be verified and updated using the SVI version that enrolled them.

To upgrade a speaker model to the latest SVI version, use the `reenrol` service. This will replace the existing speaker model with a new one based on the set of audio files that is supplied to `reenrol`. It is not possible to upgrade an existing model without supplying audio data. It is, therefore, a requirement that the original set of audio files that were used to enrol the speaker is stored for any future re-enrolments (upgrades). If the audio files are not stored, and the speaker model is to be upgraded, it will be necessary for the speaker to provide fresh audio for the re-enrolment.

Over the lifetime of a speaker model it will be regularly updated with new audio, particularly on the occasion when a verification fails and the speaker is verified by other means. These audio recordings should also be stored so that they can be included in the re-enrolment and make it current.

3 WORKING WITH THRESHOLDS

For a verification attempt to pass, the calculated “confidence” score must be greater than a threshold. In VoiSentry, each verification attempt will calculate a score, and if this score is greater than 1 the verification should be deemed successful. In this case, 1 is the threshold and it will provide a good level of security. The threshold should not be changed – see `sensitivity` below.

Please note that the algorithm that calculates the confidence for a given speaker adapts with that speaker’s model. Because of this, it cannot be said that a score of 1.2 for one speaker is better than another speaker's score of 1.1. In fact, for a given speaker, it is not possible to compare confidence scores over time (today's score of 1.2 for a given speaker is not necessarily better than yesterday's score of 1.1). Many times, a speaker will pass the verification attempt with a confidence score that is just above 1. This is normal and should not be considered a weak result.

If it becomes apparent that the error rates are not suitable for the application, a sensitivity value can be passed to the verify service to make it more or less sensitive. The sensitivity value can range from -10 to 10, by default it is 0. Setting a positive sensitivity, for example 1, will reduce the number of impostors that pass. Setting a negative sensitivity, for example -1, will reduce the number of real speakers that fail. The sensitivity mechanism provides a robust way to modify a verification attempt’s score and it should be used instead of changing the threshold.

Along with the confidence score, a value for “verified” is returned. This will be True if the verification attempt passed, False if it failed. This value should be used instead of “confidence” when checking the verification result.

4 WORKING WITH ENROLMENTS

To enrol a new speaker, call the `enrol` service with at least ten seconds of speech. The speech can be divided over more than one audio file. Calling the `enrol` service with at least ten seconds of speech will, in most cases, create a good speaker model that is ready to use. However, to properly finalise the model, the enrolment process should include two verification attempts, each followed by a call to `update`. During this process, take note of the confidence that each verification attempt returns. If the confidence is below 1, another round of verify and update should be done. The verify plus update procedure will allow the algorithm to adjust to the speaker. As a result of this, you will notice the verification confidence score getting closer to 1.

5 WORKING IN TEXT DEPENDENT MODE

A speaker is not restricted to saying the same pass phrase each time. During enrolment and verification, a speaker can say different things. However, a speaker's model will be more robust if they always speak the same utterance, and verification will be more reliable. The `verify` service has the option to be run in text dependent mode. In this mode, the verification process can make use of additional information and the confidence measure will be more informed.

Text dependent mode does not mean that all speakers must use the same phrase. Each speaker can still choose their own phrase, but they must not change it.

6 WORKING WITH UPDATES

There are a number of factors that can affect a speaker's verification confidence: updates to the model; a change of phone or environment; illness, and even simply being tired. Consequently, it is important to allow the speaker's model to adapt. A new model should be updated after each verification attempt until it has been updated two or three times (this can be part of the enrolment process); thereafter, it can be updated less frequently.

Note that that updates do not have to be done immediately after the verification session, they can be done off-line during quiet periods.

A speaker's model must only be updated with the audio from a successful verification; however, this should include verification by human interaction after a failed automatic attempt. It can happen that adverse conditions cause a speaker to fail to be verified because the confidence is too low. In these cases, a human verification process can allow the speaker to be passed, and, if the recorded speech is deemed to be of the speaker, the model can be updated. Care should be exercised in such cases because if the recording is not clean, or if the caller's identity is not correctly verified, updating the model will make it more difficult for the caller to be accepted in the future.

7 PRESENTATION ATTACK (“SPOOF”) DETECTION

If PAD is enabled the `verify` and `recognise_verify` services will return the output of a Presentation Attack Detection process (commonly known as spoof detection) that is run as part of the service. This process will try to identify audio that has been artificially produced (synthesized) or been clandestinely recorded and replayed. For the `verify` and `recognise_verify` services, it can also try to detect audio that has been used before, i.e., that is an (almost) identical copy of previously used audio.

PAD is not run during enrolment or updates. It is assumed that enrolments and updates are done in a controlled manner or environment which precludes impostor attempts.

The PAD information is returned as a JSON object with the following information:

- *type*, indicates the type of attack that may have been attempted:
  "TypeA" for duplicate audio – the same audio has been presented before;
  "TypeB" for manipulated speech or human mimicry;
  "TypeC" for synthetically generated or converted speech.
- *audio*, for "TypeA" attacks only. The names of the two audio files that were deemed duplicates. One is the current verification file, the other is historic.

The PAD object will be empty if no presentation attacks were detected, or PAD is not enabled.

8 WEB SERVICE REFERENCE

The following pages contain detailed descriptions of each web service.

8.1 ENROL

Create a speaker model and add it to the database.

Audio data can be supplied directly, or a URL to the audio can be given. Up to ten files of audio data can be supplied and each file is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM. If any URLs are given, enrol will not use any other audio sources.

The speaker model is based on the supplied audio. The audio must be a good representation of the speaker's normal speech. It is recommended that at least ten seconds of audio is provided.

The file names or URLs of the supplied audio sources are returned along with the enrolment ID and the SVI version that was used to create the speaker model. These are returned in a JSON object under the key `enrolment`.

The enrolment process should include at least two verification attempts, each followed by a call to `update`. This will allow VoiSentry to finalise the model. Take note of whether these verification attempts pass. Continue to `verify` and `update` until the verification attempts pass.

Some of the supplied audio sources may fail the initial quality analysis. These sources are also returned under the `enrolment` key as failed sources. Failed sources will not be used when creating the model and new recordings must be made.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's unique enrolment ID. A string, chosen by the application writer. Only one `enrolid` can be supplied.

Audio sources:

- **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. More than one URL can be supplied.
- **<audio sources>** - One or more handles to audio sources. If any URLs are given, enrol will not look for these.

Optional argument:

- **sviversion** - A string. The SVI version to use. Default is to use the latest version.

Usage example:

```
curl --request POST --form source1=@audio_a.wav --form source2=@audio_b.wav -k
"https://$TARGET:/api/enrol?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&sviversion=1"
```

Usage example:

```
curl -k "https://$TARGET:/api/enrol?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&sviversion=1&url=http://my.wav.files.com:8001/get_wav?filename=audi
o_a.wav"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "enrol", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "enrolment": {
      "enrolid": "99997", # A string. The ID of the newly enrolled speaker.
      "sources": ["audio_a.wav", "audio_b.wav"], # A list of strings. The names or URLs of the
audio files.
      "failed_sources": [], # A list of strings, or empty. The names or URLs of the audio
files that failed analysis.
      "svi_version": "1"
  }
}
```

8.2 VERIFY

Run a verification test on a speaker model using new audio data.

Audio data can be supplied directly, or a URL to the audio can be given. Up to ten files of audio data can be supplied and each file is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM. If any URLs are given, verify will not expect other types of audio sources.

For the verification attempt to pass, the calculated confidence must be greater than 1.

The verification results are returned in a JSON object under the key `verification`.

A verification job can be a presentation attack. Most such attacks are too crude to pass the verification attempt but more sophisticated ones could get a confidence score greater than 1. To combat these, a PAD process may be run. The result of this process is returned under the key `PAD_info` as part of the `verification` object.

The PAD process will only be run if the `pad` parameter is set to 'T', as the default is 'F', i.e. no PAD.

The results to return are:

- ***enrolid*** - A string. The enrolment ID as supplied by the user.
- ***svi_version*** - A string, the SVI version used to verify this model
- ***date*** - A string. The verification date in the format `YYYY-MM-DD_HH:MM:SS`. For example, `'2017-09-25_11:32:11'`.
- ***confidence*** - A float, a confidence measure centred around 1. A value greater than 1 should be considered a pass.
- ***verified*** - True or False. True if the verification passed, False if it failed.
- ***PAD_info*** - A JSON object. The results of the PAD (Presentation Attack Detection) process. This will be empty if no PAD attempt was detected, or PAD is disabled.
- ***sources*** - A list of strings. The audio file names or URLs
- ***failed_sources*** - A list of strings, or empty. The names or URLs of the audio files that failed analysis.

The PAD info:

- ***type*** - A string. Indicates the type of attack that may have been attempted: "TypeA" for duplicate audio (has been played before); "TypeB" for a replayed recording or imitation attempt; "TypeC" for synthetic speech.
- ***audio*** - A list of strings. For "TypeA" attacks. The names of the two audio files that were deemed duplicates. One is the current verification file, the other is historic.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's unique enrolment ID. An alphanumeric string. Only one `enrolid` can be supplied.

Optional arguments*:*

- **sensitivity** - A value between -10 and 10. This value adjusts the verification confidence measure. A negative value will increase the confidence and a positive value will decrease it.
- **pad** - To turn on/off PAD (Presentation Attack Detection). If set to “T”, the detector will be run. The default is ‘F’, i.e. to not run the detector.
- **textdependent** - To turn on or off text dependent mode. If set to "T", it is assumed that the speaker is saying the same phrase each time and that the algorithm can use additional information to calculate the score. If set to "F", it is assumed that the speaker is saying something different each time. The default is "T", text dependent mode.

Audio sources:

- **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. More than one URL can be supplied.
- **<audio sources>** - One or more handles to audio sources. If any URLs are given, verify will not look for these.

Usage example:

```
curl --request POST --form source1=@audio_a.wav --form source2=@audio_b.wav -k
"https://$TARGET:/api/verify?key=d50aae9d-424b-46b7-889b-c333059e90e4&enrolid=99997"
```

Usage example:

```
curl -k "https://$TARGET:/api/verify?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&url=http://my.wav.files.com:8001/get_wav?filename=audio_a.wav"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "verify", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "verification": {
      "enrolid": "99997", # The enrolment ID supplied by the user.
      "svi_version": "1", # A string, the SVI version used to verify this model
      "sources": ["audio_a.wav", "audio_b.wav"], # A list of strings. The names or URLs of the
audio files.
      "failed_sources": [], # A list of strings, or empty. The names or URLs of the audio
files that failed analysis.
      "confidence": 1.2, # A float, a confidence measure centred around 1.
      "verified": True, # True or False, True if the verification passed, False if it failed.
      "PAD_info": {
            "type": "TypeA",
            "audio": ["audio_a.wav", "audio_b.wav" ] }
      "date": "2017-09-25_11:32:11" # A string. The date and time as YYYY-MM-DD_HH:MM:SS.
  }
}
```

8.3 UPDATE

Update an existing speaker model with new audio data.

Audio data can be supplied directly, or a URL to the audio can be given. Up to ten files of audio data can be supplied and each file is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM. If any URLs are given, update will not use any other audio sources. More than one audio file can be supplied.

It is important to allow the speaker's model to adapt over time and new models should be updated regularly. It is good practice to update the speaker's model after each successful verification attempt, using the audio that was used for the successful verification.

The file names or URLs of the supplied audio sources are returned along with the enrolment ID. These are returned in a JSON object under the key `updated`.

Some of the supplied audio sources may fail the analysis. These sources are also returned under the `updated` key.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's unique enrolment ID. An alphanumeric string. Only one `enrolid` can be supplied.

Audio sources:

- **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. More than one URL can be supplied.
- **<audio sources>** - One or more handles to audio sources. If any URLs are given, update will not look for these.

Usage example:

```
curl --request POST --form source1=@audio_a.wav --form source2=@audio_b.wav -k
"https://$TARGET:/api/update?key=d50aae9d-424b-46b7-889b-c333059e90e4&enrolid=99997"
```

Usage example:

```
curl -k "https://$TARGET:/api/update?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&url=http://my.wav.files.com:8001/get_wav?filename=audio_a.wav"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "update", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "updated": {
      "enrolid": "99997", # A string. The ID of the newly enrolled speaker.
      "sources": ["audio_a.wav", "audio_b.wav"] # A list of strings. The names or URLs of the
audio files.
      "failed_sources": [], # A list of strings, or empty. The names or URLs of the audio
files that failed analysis.
      "svi_version": "1"
  }
}
```

8.4 IDENTIFY

Calculate a confidence score for each of the supplied enrolment ids. Return a sorted list of the confidences, from most confident to least confident.

Audio data can be supplied directly, or a URL to the audio can be given. Up to ten files of audio data can be supplied and each file is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM. If any URLs are given, identify will not use any other audio sources.

The confidences are returned in a JSON object under the key `identification`.

The results to return are:

- ***svi_versions*** - A list of strings, the SVI versions used to identify the models.
- ***date*** - A string. The identification date in the format `YYYY-MM-DD_HH:MM:SS`. For example, `'2017-09-25_11:32:11'`.
- ***confidences*** - A sorted list of pairs. Sorted by confidence. Each pair is a list containing [confidence, enrolment ID].

- ***sources*** - A list of strings. The audio file names or URLs.
- ***failed_sources*** - A list of strings, or empty. The names or URLs of the audio files that failed analysis.

The speaker model that returns the highest confidence will be first in the list. If the confidence is below 1, this might indicate that none of the enrolment Ids match the supplied audio data.

Required arguments*:*

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's enrolment ID. An alphanumeric string. More than one `enrolid` can be supplied.

- **Audio sources*:***
  - **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. More than one URL can be supplied.
  - **<audio sources>** - One or more handles to audio sources. If any URLs are given, identify will not look for these.

- **Optional arguments*:***
  - **sensitivity** - A value between -10 and 10. This value adjusts the verification confidence measure. A negative value will increase the confidence and a positive value will decrease it.
  - **number** - An integer. The number of confidences to return. The default is to return a confidence for each enrolment ID supplied.

Usage example:

```
curl --request POST --form source1=@audio_a.wav -k
"https://$TARGET:/api/identify?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&enrolid=99998&number=2"
```

Usage example:

```
curl -k "https://$TARGET:/api/identify?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&enrolid=99998&number=2&url=http://my.wav.files.com:8001/get_wav?fil
ename=audio_a.wav"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "identify", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "identification" : {
      "svi_versions": ["1",], # A list of strings, the SVI versions used to identify the
models.
      "sources": ["audio_a.wav", "audio_b.wav"], # A list of strings. The names or URLs of the
audio files.
      "failed_sources": [], # A list of strings, or empty. The names or URLs of the audio
files that failed analysis.
      "confidences": [[1.28, 9998], [1.23, 9997]] # A sorted list of pairs. Sorted by
confidence. Each pair is a [confidence, enrolment ID] list.
      "date": "2017-09-25_11:32:11" # A string. The date and time as YYYY-MM-DD_HH:MM:SS.
  }
}
```

8.5 REENROL

Use this command to remove an existing model and replace it with a new one.

Audio data can be supplied directly, or a URL to the audio can be given. Up to ten files of audio data can be supplied and each file is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM. If any URLs are given, reenrol will not use any other audio sources.

An update to the system may result in a new SVI version. Any existing speaker models will continue to use the SVI version that created them - this avoids any incompatibility issues. For a speaker model to use a later SVI version the speaker must be re-enrolled.

To re-enrol a speaker, it is advised that the original set of audio files that were used to enrol the speaker is supplied. Over the lifetime of a speaker model it will be regularly updated with new audio, particularly on the occasion when verification fails and the speaker is verified by other means. These audio recordings should also be stored so that the new model can be further updated and made current.

The file names or URLs of the supplied audio sources are returned along with the enrolment ID and the SVI version used. These are returned in a JSON object under the key `enrolment`.

Some of the supplied audio sources may fail the analysis. These sources are also returned under the `enrolment` key.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's unique enrolment ID. An alphanumeric string. Only one `enrolid` can be supplied.

Audio sources:

- **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. More than one URL can be supplied.
- **<audio sources>** - One or more handles to audio sources. If any URLs are given, reenrol will not look for these.

Optional argument:

- **sviversion** - A string. The SVI version to use. Default is to use the latest version.

Usage example:

```
curl --request POST --form source1=@audio_a.wav --form source2=@audio_b.wav -k
"https://$TARGET:/api/reenrol?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&sviversion=1"
```

Usage example:

```
curl -k "https://$TARGET:/api/reenrol?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&sviversion=1&url=http://my.wav.files.com:8001/get_wav?filename=audi
o_a.wav"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "reenrol", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "enrolment" : {
      "enrolid": "99997", # A string. The ID of the newly enrolled speaker.
      "sources": ["audio_a.wav", "audio_b.wav"], # A list of strings. The names or URLs of the
audio files.
      "failed_sources": [], # A list of strings, or empty. The names or URLs of the audio
files that failed analysis.
      "svi_version": "1"
  }
}
```

8.6 RECOGNISE

Run a speech recognition task on some audio data. Speech recognition can be used to recognise the digits zero to nine or the words yes and no. Five languages are supported, German, English, French, Spanish and Italian.

Audio data can be supplied directly, or a URL to the audio can be given. Only one set of audio data can be supplied and it is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM. If a URL is given, recognise will not use any other audio sources.

DTMF digit detection can be run on the audio together with speech recognition.

If words are recognised, they are returned in a list of strings, for example, ['one', 'two', 'three', 'four']. If DTMF digits are recognised, they are returned as a string, for example, '1234'.

The arguments are described below.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **type** - The speech recognition grammar type to use. A string. Only one grammar type can be specified. See below for the list of grammar options.

Audio sources:

- **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. Only one can be supplied.
- **<audio sources>** - An audio source handle. If a URL is given, recognise will not look for this. Only one can be supplied.

Optional arguments:

- **enrolid** - The target speaker's enrolment ID. An alphanumeric string. Only one `enrolid` can be supplied.
- **checkdtmf** - A character. If set to `'y'`, DTMF digit detection will be attempted on the audio signal. DTMF detection will be attempted before speech recognition is run. If DTMF digits are detected, speech recognition is not attempted.

Recognition grammars:

- **Deu1digit** - German, recognise a single digit.
- **Deu2digits** - German, recognise two digits.
- **Deu3digits** - German, recognise three digits.
- **Deu4digits** - German, recognise four digits.
- **DeuNdigits** - German, recognise any number of digits.
- **DeuYesNo** - German, recognise the words `ja` or `nein`.
- **Eng1digit** - English, recognise a single digit.
- **Eng2digits** - English, recognise two digits.

- **Eng3digits** - English, recognise three digits.
- **Eng4digits** - English, recognise four digits.
- **EngNdigits** - English, recognise any number of digits.
- **EngYesNo** - English, recognise the words `yes` or `no`.
- **Fra1digit** - French, recognise a single digit.
- **Fra2digits** - French, recognise two digits.
- **Fra3digits** - French, recognise three digits.
- **Fra4digits** - French, recognise four digits.
- **FraNdigits** - French, recognise any number of digits.
- **FraYesNo** - French, recognise the words `oui` or `non`.
- **Ita1digit** - Italian, recognise a single digit.
- **Ita2digits** - Italian, recognise two digits.
- **Ita3digits** - Italian, recognise three digits.
- **Ita4digits** - Italian, recognise four digits.
- **ItaNdigits** - Italian, recognise any number of digits.
- **ItaYesNo** - Italian, recognise the words `si` or `no`.
- **Spa1digit** - Spanish, recognise a single digit.
- **Spa2digits** - Spanish, recognise two digits.
- **Spa3digits** - Spanish, recognise three digits.
- **Spa4digits** - Spanish, recognise four digits.
- **SpaNdigits** - Spanish, recognise any number if digits.
- **SpaYesNo** - Spanish, recognise the words `si` or `no`.

The recognition result is returned as a JSON object under the key `recognition`. The recognition object will contain:

- ***words*** - A list of strings, e.g., `['one', 'two', 'three', 'four']`. This key will be omitted if DTMF digits were detected.
- ***dtmf*** - A string of digits, e.g., `'1234'`. DTMF detection is run if `checkdtmf` is set to `'y'`. If DTMF digits are found, this will replace the `'words'` key.
- ***type*** - A string. The recognition type that was requested, e.g., `'4digits'`.
- ***source*** - The file name or URL of the supplied audio source.
- ***date*** - A string. The recognition date and time in the format `YYYY-MM-DD_HH:MM:SS`. For example, `'2017-09-25_11:32:11'`.

If neither `words` nor `dtmf` were recognised, the `recognition` key will be omitted from the result.

Usage example:

```
curl --request POST --form source1=@audio_a.wav -k
"https://$TARGET:/api/recognise?key=d50aae9d-424b-46b7-889b-
c333059e90e4&type=Eng4digits&checkdtmf=y"
```

Usage example:

```
curl -k "https://$TARGET:/api/recognise?key=d50aae9d-424b-46b7-889b-
c333059e90e4&type=Eng4digits&checkdtmf=y&url=http://my.wav.files.com:8001/get_wav?filename=aud
io_a.wav"
```

If `checkdtmf` is set to `'y'`, this function will detect any DTMF digits present in the audio recording. If DTMF digits are found, ASR will not be run on the recording.

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "recognise", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "recognition": {
      "source": "audio_a", # The name or URL of the audio file that was supplied.
      "words": ["one", "two", "three", "four"], # A list of strings. Omitted if DTMF is
detected.
      "dtmf": "1234", # If checkdtmf was set, DTMF digit detection will be run.
      "type": "4digits", # A string. The grammar type that was used.
      "date": "2017-09-25_11:32:11" # A string, the date and time as YYYY-MM-DD_HH:MM:SS.
  }
}
```

8.7 RECOGNISE_VERIFY

Run a speech recognition and verification task on some audio data. Speech recognition can be used to recognise the digits zero to nine or the words yes and no. Five languages are supported, German, English, French, Spanish and Italian.

Audio data can be supplied directly, or a URL to the audio can be given. Only one set of audio data can be supplied and it is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM. If a URL is given, recognise_verify will not use any other audio sources.

The speech recognition results and the verification results are returned, as they are for the `recognise` and `verify` services. Unlike the `recognise` service, DTMF digit recognition cannot be set for this service.

The words that are recognised are returned in a list of strings, for example, ['one', 'two', 'three', 'four'].

The arguments are described below.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's enrolment ID. An alphanumeric string. Only one ID can be supplied.
- **type** - The speech recognition grammar type to use. A string. Only one grammar type can be specified. See below for the list of grammar options.

Optional arguments:

- **sensitivity** - A value between -10 and 10. This value adjusts the verification confidence measure. A negative value will increase the confidence and a positive value will decrease it.
- **pad** - To turn on/off PAD (Presentation Attack Detection). If set to “T”, the detector will be run. The default is ‘F’, i.e. do not run the detector.
- **textdependent** - To turn on or off text dependent mode. If set to "T", it is assumed that the speaker is saying the same phrase each time and that the algorithm can use additional information to calculate the score. If set to "F", it is assumed that the speaker is saying something different each time. The default is "T", text dependent mode.

Audio sources:

- **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. Only one can be supplied.
- **<audio sources>** - An audio source handle. If a URL is given, recognise_verify will not look for this. Only one can be supplied.

Recognition grammars:

- **Deu1digit** - German, recognise a single digit.
- **Deu2digits** - German, recognise two digits.
- **Deu3digits** - German, recognise three digits.
- **Deu4digits** - German, recognise four digits.
- **DeuNdigits** - German, recognise any number of digits.
- **DeuYesNo** - German, recognise the words `ja` or `nein`.
- **Eng1digit** - English, recognise a single digit.
- **Eng2digits** - English, recognise two digits.
- **Eng3digits** - English, recognise three digits.
- **Eng4digits** - English, recognise four digits.
- **EngNdigits** - English, recognise any number of digits.
- **EngYesNo** - English, recognise the words `yes` or `no`.
- **Fra1digit** - French, recognise a single digit.
- **Fra2digits** - French, recognise two digits.
- **Fra3digits** - French, recognise three digits.
- **Fra4digits** - French, recognise four digits.
- **FraNdigits** - French, recognise any number of digits.
- **FraYesNo** - French, recognise the words `oui` or `non`.
- **Ita1digit** - Italian, recognise a single digit.
- **Ita2digits** - Italian, recognise two digits.
- **Ita3digits** - Italian, recognise three digits.

- **Ita4digits** - Italian, recognise four digits.
- **ItaNdigits** - Italian, recognise any number of digits.
- **ItaYesNo** - Italian, recognise the words `si` or `no`.
- **Spa1digit** - Spanish, recognise a single digit.
- **Spa2digits** - Spanish, recognise two digits.
- **Spa3digits** - Spanish, recognise three digits.
- **Spa4digits** - Spanish, recognise four digits.
- **SpaNdigits** - Spanish, recognise any number if digits.
- **SpaYesNo** - Spanish, recognise the words `si` or `no`.

The recognition result is returned as a JSON object under the key `recognition`. The recognition object will contain:

- ***words*** - A list of strings, e.g., `['one', 'two', 'three', 'four']`.
- ***type*** - A string. The recognition type that was requested, e.g., `'4digits'`.
- ***source*** - A string. The file name of the supplied audio source.
- ***date*** - A string. The recognition date and time in the format `YYYY-MM-DD_HH:MM:SS`. For example, `'2017-09-25_11:32:11'`.

If no `words` were recognised, the `recognition` key will be omitted from the result.

The verification results are returned in a JSON object under the key `verification`. The results to return are:

- ***enrolid*** - A string. The enrolment ID as supplied by the user.
- ***svi_version*** - A string, the SVI version used to verify this model
- ***confidence*** - A float, a confidence measure centred around 1. A value greater than 1 should be considered a pass.
- ***PAD_info*** – A JSON object. The results of the PAD (Presentation Attack Detection) process. This will be empty if no PAD attempt was detected, or PAD is disabled.
- ***verified*** - True or False. True if the verification passed, False if it failed.
- ***source*** - A string. The file name or URL of the supplied audio source.
- ***date*** - A string. The verification date in the format `YYYY-MM-DD_HH:MM:SS`. For example, `'2017-09-25_11:32:11'`.

The PAD info:

- ***type*** - A string. Indicates the type of attack that may have been attempted: "TypeA" for duplicate audio (has been played before); "TypeB" for replayed audio or an imitation attempt; "TypeC" for synthetically constructed audio.
- ***audio*** - A list of strings. For the "TypeA" attack. The names of the two audio files that were deemed duplicates. One is the current verification file, the other is historic.

Usage example:

```
curl --request POST --form source1=@audio_a.wav -k
"https://$TARGET:/api/recognise_verify?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&type=Eng4digits"
```

Usage example:

```
curl -k "https://$TARGET:/api/recognise_verify?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&type=Eng4digit&url=http://my.wav.files.com:8001/get_wav?filename=au
dio_a.wav"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "recognise_verify", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "recognition': {
      "source": "audio_a", # The name of the audio file that was supplied.
      "words": ["one", "two", "three", "four"], # A list of strings.
      "type": "4digits", # A string. The grammar type that was used.
      "date": "2017-09-25_11:32:11" # A string, the date and time as YYYY-MM-DD_HH:MM:SS.
  },
  "verification": {
      "enrolid": "99997", # The enrolment ID supplied by the user.
      "svi_version": "1", # A string, the SVI version used to verify this model
      "source": "audio_a.wav", # A string. The name or URL of the supplied audio file.
      "failed_source": "" # A string (can be empty). The name or URL of the audio file that
failed analysis.
      "confidence": 1.2, # A float, a confidence measure centred around 1.
      “verified”: True, # True or False, True if the verification passed, False if it failed.
      "PAD_info": {
            "type": "TypeA",
            “audio": ["audio_a.wav", "audio_b.wav" ]
      }
      "date": "2017-09-25_11:32:11" # A string. The date and time as YYYY-MM-DD_HH:MM:SS.
  }
}
```

8.8 SIMILARITY

Run a similarity test on one or more pairs of audio data.

Audio data can be supplied directly, or a URL to the audio can be given. At least two files of audio data must be supplied and each file is limited to sixty seconds. The audio data must be mono WAV files. The minimum bit depth is 16 (unless the format is A-Law or mu-Law). The preferred (also the minimum allowed) sampling rate is 8 kHz and the preferred format is 16-bit PCM.

This method is provided to help determine whether one audio source is a copy of the other. Some presentation attacks may use a clandestinely recorded instance of a legitimate verification attempt. The recording will probably not be identical to the original, but it will be very similar.

If there is a suspicion that a verification attempt may be an attack, this method can be used to compare one or more stored known genuine attempts with the suspected attacks. If a suspected attack is a copy of one of the stored recordings, this method will return a high similarity score for that pair.

Each of the audio files provided is compared with the others and all the similarity scores are returned along with the corresponding file name pairs.

The similarity score returned will be a value between -5 and 5. If a value greater than 2 is returned, the two audio sources are very similar and one is probably a copy of the other. A value greater than 3.5 means that the two audio sources can be considered identical.

The results to return are:

- ***similarity*** - A list of pairs. Each pair is a list of two items. The first item is the similarity score, the second is a list containing the two file names.
- ***sources*** - A list of strings. The audio file names or URLs.

Audio sources*:*

- **url** - A string. The address from which to download audio data. For example `http://my.wav.files.com:8001/get_wav?filename=audio_a.wav`. If using URLs, two or more must be supplied.
- **<audio sources>** - Two or more handles to audio sources. If URLs are given instead, this method will not look for these.

Usage example:

```
curl --request POST --form source1=@audio_a.wav --form source2=@audio_b.wav --form
source2=@audio_c.wav -k "https://$TARGET:/api/similarity"
```

Usage example:

```
curl -k
"https://$TARGET:/api/similarity?url=http://my.wav.files.com:8001/get_wav?filename=audio_a.wav
&url=http://my.wav.files.com:8001/get_wav?filename=audio_b.wav"
&url=http://my.wav.files.com:8001/get_wav?filename=audio_c.wav"
```

Returns:

```
{
  "transactionid" : "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "similarity": [[0.8938, ["audio_a.wav", "audio_b.wav"]], [0.0188, ["audio_a.wav",
"audio_c.wav"]], [-1.7901, ["audio_b.wav", "audio_c.wav"]]]
  "sources": ["audio_a.wav", "audio_b.wav", "audio_c.wav"] # A list of strings. The names or
URLs of the audio files.
}
```

8.9 PING

A simple ping interface.

The ping information returned is:

- *clustername* - A string. The name given to this cluster.
- *serialno* - A string. This node's serial number.
- *userip* - A string. The IP address from which the ping request originated.
- *nodeactive4* - A list. IPv4 addresses of active nodes with IPv4 addresses.
- *nodeactive6* - A list. IPv6 addresses of active nodes with IPv6 addresses.
- *buildversion* - A string. The VoiSentry build version.

The ping information is returned in a JSON object under the key `pinginfo`.

Required argument:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.

Usage example:

```
curl -k https://$TARGET:/api/ping?key=d50aae9d-424b-46b7-889b-c333059e90e4
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "ping", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "pinginfo": {
      "clustername": "cluster", # A string, the cluster name.
      "serialno": "1499263460", # A string, the current node's serial number.
      "userip": "10.100.100.10", # A string, the user's IP address.
      "nodeactive4": ['10.11.12.13'.'10.22.12.14'], # A list of IPv4 addresses of active nodes
with IPv4 addresses.
      "nodeactive6": [], # A list of IPv6 addresses of active nodes with IPv6 addresses.
      "buildversion": "1.17.8" # The VoiSentry build version.
  }
}
```

8.10 QUOTAS

Retrieve the user's accesskey quotas and current counts.

The quotas and counts to return are:

- *maxenrols* - The maximum number of enrolments allowed using this accesskey. 0 means unlimited.
- *maxverifs* - The maximum number of verifications per day allowed using this accesskey. 0 means unlimited.
- *maxidents* - The maximum number of identifications per day allowed using this accesskey. 0 means unlimited.
- *currenrols* - The current enrolment count.
- *currverifs* - The current number of verifications for this day.
- *curridents* - The current enrolment count.

The quotas and counts are returned in a JSON object. The object key in the results object is `quotas`.

Required argument:

- **key** - The user access key.

Usage example:

```
curl -k "https://$TARGET:/api/quotas?key=2fed35b8-985e-44c4-9a25-19cbfa480ce7"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "quotas", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "quotas" : {
      "maxenrols": 100, # An integer, the maximum number of enrolments per day.
      "maxverifs": 1000, # An integer, the maximum number of verifications per day.
      "maxidents": 1000, # An integer, the maximum number of identifications per day.
      "currenrols": 50, # An integer, the number of enrolments performed, so far, this day.
      "currverifs": 500, # An integer, the number of verifications performed, so far, this
day.
      "curridents": 50 # An integer, the number of identifications performed, so far, this
day.
  }
}
```

8.11 QUERY

Return some verification statistics for one or more speakers.

The statistics to return are:

- *deletion_date* - A string. The date and time the speaker's model was deleted. The format is `YYYY-MM-DD_HH:MM:SS` or an empty string if not deleted.
- *update_date* - A string. The date and time the speaker's model was updated. The format is `YYYY-MM-DD_HH:MM:SS` or an empty string if not updated.
- *enrolment_date* - A string. The date and time the speaker first enrolled. The format is `YYYY-MM-DD_HH:MM:SS`. For example, `'2017-09-25_11:32:11'`.
- *verification_attempts* - The number of times the speaker has attempted to verify. An integer.
- *verifications_passed* - The number of successful verification attempts. An integer - see below.
- *verifications_failed* - The number of unsuccessful verification attempts. An integer - see below.
- *last_verification_date* - A string. The date and time of the last verification attempt. The format is `YYYY-MM-DD_HH:MM:SS`. For example, `'2017-09-25_11:32:11'`.
- *model_version* - The current speaker model version number. An integer. This is incremented after each model update (not to be confused with re-enrolments).
- *svi_version* - The SVI version used to create this model. A string. To upgrade to a later version, run a re-enrolment.
- *original_sources* - The source names used for the original enrolment.

`verifications_passed` and `verifications_failed` are calculated using the threshold supplied to the verify service, or the default threshold if none was supplied.

Since the verify service does return the confidence measure, the application may choose to pass or fail an individual verification based on a different criterion - ignoring the comparison with the supplied (or default) threshold.

However, the `verifications_passed` and `verifications_failed` counters are unable to reflect those decisions.

The statistics are returned as a JSON object containing the keys shown above. One such object is returned for each enrolment id supplied. The enrolment ids will be found under the key `statistics` in the returned object.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's enrolment ID. An alphanumeric string. More than one enrolment ID can be provided. For example `'99997'`.

Usage example:

```
curl -k "https://$TARGET:/api/query?key=73c1d0b1-9ffc-4696-8677-
31349a716a15&enrolid=99997&enrolid=99998"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "query", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "statistics": {
      "99997": {
          "update_date": "", # A string, YYYY-MM-DD_HH:MM:SS, empty if enrolment is not
updated.
          "deletion_date": "", # A string, YYYY-MM-DD_HH:MM:SS, empty if enrolment is not
deleted.
          "enrolment_date": "2017-09-25_11:32:11", # A string, YYYY-MM-DD_HH:MM:SS.
          "model_version": 5, # An integer, incremented with each model update.
          "svi_version": "1",# A string, the SVI version used to create this model
          "verification_attempts": 10, # An integer, the total number of verification
attempts.
          "last_verification_date': "", # A string, YYYY-MM-DD_HH:MM:SS, empty is no
verification has been done.
          "verifications_passed": 5, # An integer, the number of verification passes.
          "verifications_failed": 5, # An integer, the number of verification failures.
           "original_sources": ["source1", "source2"] # A list of strings, the original
enrolment sources
      },
      "99998": { ... }, # As above
  }
}
```

8.12 CHECK

Supply one or more enrolment IDs. Matching IDs currently enrolled in the system are returned. IDs that are not found (or have been deleted) are omitted from the list.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The enrolment ID to check. An alphanumeric string. More than one enrolment ID can be provided.

Usage example:

```
curl -k "https://$TARGET:/api/check?key=73c1d0b1-9ffc-4696-8677-
31349a716a15&enrolid=99997&enrolid=99998"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "check", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "enrolids": ["99997", "99998"] # A list of strings. Matching IDs enrolled in the system.
}
```

8.13 DELETE

Remove one or more speaker enrolments from the database.

Supply one or more enrolment IDs. The list of enrolment IDs that were removed is returned.

Enrolment IDs that are not found, or are already deleted, will be omitted from the returned list.

Deleted enrolments are not physically removed, just marked as deleted. The removed enrolment IDs can be used for a different speaker, in which case the existing data is replaced.

Required arguments:

- **key** - The user access key. A string. For example, `'73c1d0b1-9ffc-4696-8677-31349a716a15'`.
- **enrolid** - The target speaker's enrolment ID. An alphanumeric string. More than one enrolment ID can be provided.

Usage example:

```
curl -k "https://$TARGET:/api/delete?key=d50aae9d-424b-46b7-889b-
c333059e90e4&enrolid=99997&enrolid=99998"
```

Returns:

```
{
  "dataset": "dataset1", # A string, the dataset referenced by the current key.
  "tenant": "tenant1", # A string, the tenant referenced by the current key.
  "service": "delete", # A string, the service that was run.
  "transactionid": "a7e1b3f6a1cd11e78439000c29c5390d", # A string, the unique transaction ID.
  "diskalert": True / False, # A bool, True if disk usage has reached the alert level.
  "memfree": 50,# An integer, the amount of free memory as a percentage.
  "nodestatus": "A", # 'A' active, 'B' blocked (the node may be going out of service).
  "enrolids": ["99997", "99998"] # A list of strings. The enrolment IDs that were deleted.
}
```

Follow VoiSentry: Twitter | Blog | LinkedIn

+44 (0) 1908 273 800 | www.voisentry.com

aculab