Smart Cloud Communications

Realtime API

API Version: 1.0.0
Document Revision: 17
Last change: 21 august 2019

Kwebbl BV

Modemstraat 1
1033RW AMSTERDAM
Netherlands

info@kwebbl.com
www.kwebbl.com

Table of Contents

Introduction

Intended audience

This document describes the technical specifications of the Realtime API and is only intended to be used by Software Engineers aiming to implement the Realtime API into a third party application.

Example use cases

The Realtime API brings a bunch of events and call actions that will allow flexible integrations with 3rd party and custom software. Events include call being started, answered, transferred, recorded etcetera whilst actions include starting calls, stopping calls and transferring calls.

Use cases that could be created with this API include:

- Call me back on an ecommerce site
- Customer Card popups in CRM
- Automatically open tickets in an inbound support call center
- Show the BLF status of all co-workers in an online company guide
- Click to dial from an online company phone book
- *Etcetera….*

Terms of Use

1. **Purpose**
   The Kwebbl Realtime API is an open platform. You can access this infrastructure free of charge, provided you share our goals, beliefs and adhere to these terms and disclaimers as laid out below.
2. **Data Privacy and Security**
   a. Your network, operating system and the software of your web servers, databases, and computer systems (collectively, "Systems") must be properly configured to securely operate your Application and store Content. Your Application must use reasonable security measures to protect the private information of your users. You must not architect or select Systems in a manner to avoid the foregoing obligation.
   b. You must promptly report any security deficiencies in, or intrusions to, your Systems that you discover to Kwebbl in writing via email to support@kwebbl.com. You will work with Kwebbl to immediately correct any security deficiency, and will immediately disconnect any intrusions or intruder. In the event of any security deficiency or intrusion involving the Application or APIs, you will make no public statements (e.g. press, blogs, social media, bulletin boards, etc.) without prior written and express permission from Kwebbl in each instance.
   c. Once you start using the API you will be given Access Credentials for your Application. "Access Credentials" means the necessary security keys, secrets, tokens, and other credentials to access the APIs. All activities that occur using your Access Credentials are your responsibility. Keep them secret. Do not sell, transfer, or sublicense them. Loss or theft of these Access Credentials could lead to high cost due to fraudulent calls, for which you will be held responsible and liable.
3. **Monetization and Fees**
   a. You are not allowed to monetize the applications that you build on top of the APIs without prior approval from Kwebbl.
   b. The APIs are currently provided free of charge, but Kwebbl reserves the right to charge for the APIs in the future. If we do charge a fee for use of the APIs or any developer tools and features, you do not have any obligation to continue to use our developer resources.

4. **Support and Modifications**
    a. We may provide you with support or modifications for the APIs in our sole discretion and we may stop providing support or modifications to you at any time without notice or liability to you.
    b. We may release subsequent versions of the APIs and require that you use those subsequent versions. Your continued use of the APIs following a subsequent release will be deemed your acceptance of modifications.
5. **Monitoring**
   You agree to assist Kwebbl in verifying your compliance with these Terms by providing us with information about your Application and storage of Content, which may also include access to your Application and other materials related to your use of the APIs. If you do not demonstrate full compliance with these Terms, we may restrict or terminate your access to the APIs.
6. **Term, Suspension and Termination**
    a. The term of these Terms will commence on the date upon which you agree to these Terms and will continue until terminated. You may terminate these Terms by discontinuing use of our APIs.
    b. We may suspend or terminate your use of all or any of the APIs at any time if we believe you have violated these Terms or if we believe the availability of the APIs in your Application is not in our or our clients' best interests.
    c. We may discontinue the availability of some or all of the APIs at any time for any reason. We may also impose limits on certain features and services or restrict your access to some or all of the APIs. All of our rights in these Terms may be exercised without prior notice or liability to you.
7. **Compliance and Amendments to these Terms**
    a. You must comply with these Terms in order to use the APIs.
    b. We reserve the right to modify, supplement, or replace the terms of these Terms, effective prospectively upon posting on our Support Site or otherwise notifying you. For example, we may publish an article on the Support Site when we have amended these Terms so that you may access and review the changes prior to your continued use of the APIs and Developer Resources. If you do not want to agree to changes to these Terms, you can terminate these Terms at any time by discontinuing use of our APIs.
8. **Disclaimer**

a. We provide this documentation, the API and all other resources on an "as is" and "as available" basis with no warranties, either express or implied, of any kind.
b. Kwebbl does not guarantee that the APIs, related services or any other developer resources it provides will function without interruption or errors in functioning. In particular, the operation of the Kwebbl APIs may be interrupted due to maintenance, updates or system or network failures. Kwebbl disclaims all liability for damages caused by any such interruption or errors in functioning.
c. Furthermore, Kwebbl disclaims all liability for any malfunctioning, impossibility of access, or poor use condition of the Kwebbl APIs due to inappropriate equipment, disturbances related to Internet Service Providers, to the saturation of the Internet Network or for any other reason.

Getting Started

Prerequisites

Connecting with the API requires you to have a webserver available for public access. For security reasons it is required that this webserver is available using SSL/HTTPS, the certificate may however be self-signed. Client applications will require a server side http component to mediate in the oAuth2 authentication and to receive call events.

Implementation of this API will require knowledge of software development and common concepts like REST, CRUD, Webhooks and oAuth 2.

The Basics

Protocol	HTTPS
Host	realtime.api.kwebbl.net *Note: Custom URL for whitelabel clients! Eg realtime.api.yourdomain.com*
Base URL	/v100/
Content Type	JSON
Authorization Standard	oAuth 2, 3-legged with Bearer type tokens RFC6749 RFC6750
Common Concepts	REST CRUD Webhooks (subscriptions)

Authorization

Overview

Each request to resources of the Realtime API must be authorized, for this we use the 3-legged Authorization Code Grant Flow of the industry standard OAuth 2.0 protocol.

Definitions

Resource Owner (user)	The entity that is in control of the data exposed by the API, e.g. a company user
Client Application	The mobile app, web site, etc. that wants to access data on behalf of the Resource Owner
Authorization Server (AS)	The Security Token Service (STS) or, colloquially, the OAuth server that issues tokens
Resource Server (RS)	The service that exposes the data, i.e. the API
Client ID	Provided after installation/creation of the application in the Company Panel
Client Secret	Provided after installation/creation of the application in the Company Panel
Redirect URL	The callback URL of the client application. It is the same as filled in during installation/creation of the application in the Company Panel.

Authorization Code Grant Flow

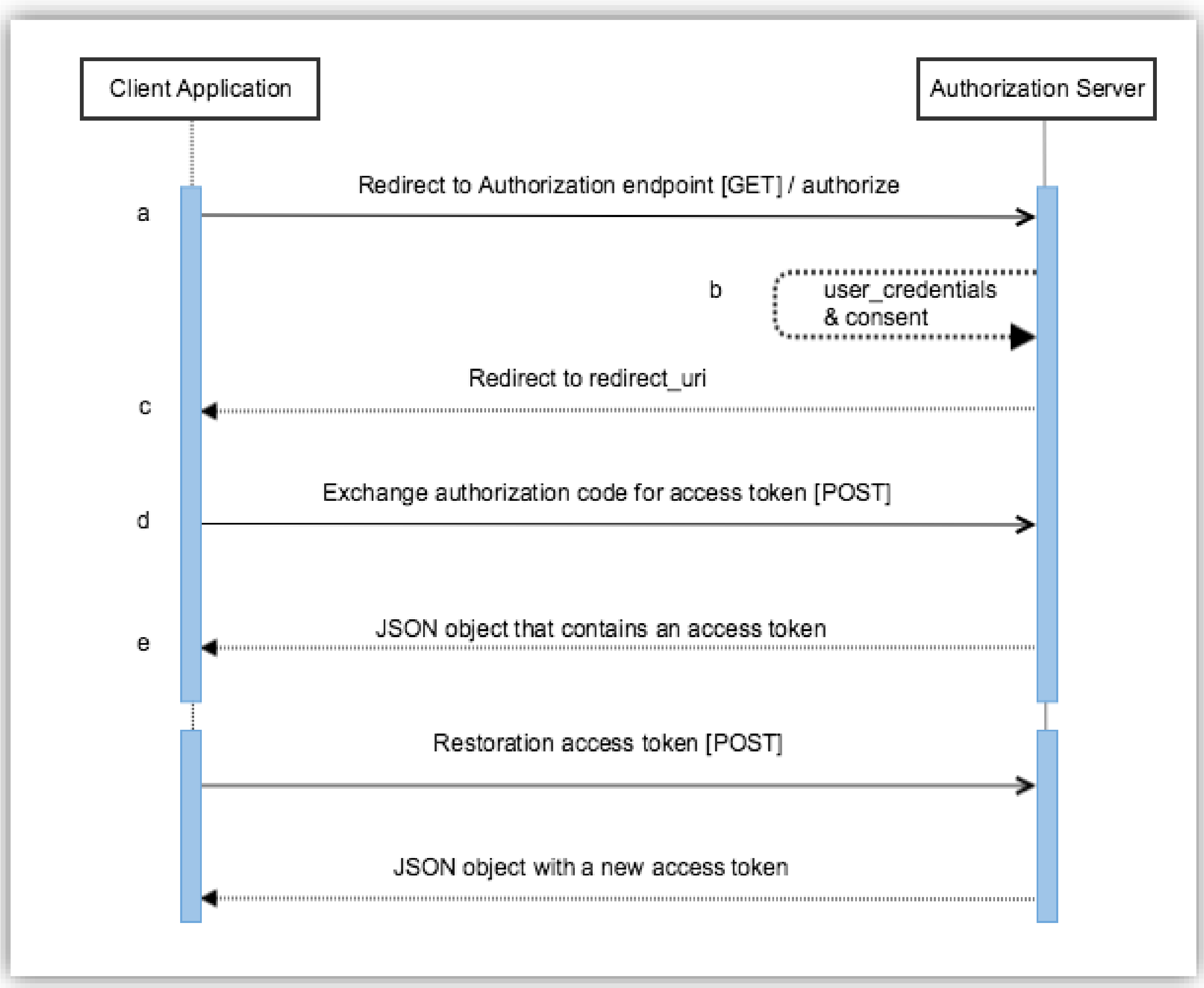
Client Application
Authorization Server
a
Redirect to Authorization endpoint [GET] / authorize
b
user_credentials
& consent
Redirect to redirect_uri
c
Exchange authorization code for access token [POST]
d
JSON object that contains an access token
e
Restoration access token [POST]
JSON object with a new access token

Figure 1 An overview of the steps involved in obtaining authorization

The Authorization Code flow is used when a Client Application (a third-party server or web application) would like to obtain access to a protected resource. The Client Application does not require access to any credentials of the resource owner (user), but instead obtains a key/token to access the protected resource on behalf of the resource owner.

Getting access to the protected resource involves the following steps:

a) Redirect the user to the Authorization endpoint

When user first tries to perform an action that requires authentication, the Client Application will forward the user the following endpoint:

```
GET /authorize
```

The following query parameters are accepted by this endpoint:

Parameter	Type	Description
client_id	string	Required. The client ID provided by company admin after installation the application in Panel
redirect_uri	string	Required. The callback URL of the client application
response_type	string	Required. The response type, expected from authorization endpoint. Value: "code"
state	string	Optional. A random string. It is used to protect against cross-site request forgery attacks.
scope	string	Optional. The scope of the access request, represented as a list of space-delimited, case-sensitive strings.

Example URL:

```
GET
/authorize?response_type=code&client_id=s6BhdRkqt3&state=xyz&redirect_uri=htt
ps%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb
```

If the request to authorization endpoint fails due to invalid parameters (e.g. id, callbacks etc.), the Authorization Server will inform Resource Owner (user) about the error but will not redirect back to client callback.

b) User authentication and granting permissions

After redirecting the user, the Authorization server will prompt the Resource Owner (user) to enter his login credentials.
After a successful login the user gives permission to Client application to access his resources and the requested scopes.

c) Redirection to client application with authorization code

The Authorization server will redirect the user back to the redirect_uri that was specified in step a.

Access granted

If the user granted access to client application, the Authorization server appends an additional query parameter **CODE** to the redirect_uri. This parameter contains the (temporary) authorization code that Client application can exchange for an access token in step d.

```
GET {redirect_uri}?code={code}
```

Access not granted

If the user did not grant access to the client application, the Authorization server appends an additional query parameter **ERROR** to the redirect_uri. This parameter contains an error code that explains why access was denied.

```
{redirect_uri}?error={error_code}
```

If the user entered invalid credentials, the client application will receive the error **INVALID_GRANT**.

```
{redirect_uri}?error=invalid_grant
```

If the user refused to grant access to client application, the Authorization server will send error code **ACCESS_DENIED**.

```
{redirect_uri}?error=access_denied
```

The request from to the Client application may contain the following query parameters:

Parameter	Type	Description
error	string	Required. Error code
error_description	string	Optional. Text with additional information about error
state	string	Required if it was present in the client authorization request

Possible error codes are:

Error code	Description
invalid_request	Request is missing a required parameter or its value is invalid
unauthorized_client	Client cannot use this authorization method
access_denied	User denied the permission grant request
unsupported_response_type	The response_type of request is not supported
server_error	Authorization server encountered an unexpected error

Client state

If the optional **STATE** parameter was passed to authorization endpoint, the client application will receive its value in the callback and should validate it against step a.

```
{redirect_uri}?code={code}&state={state}
{redirect_uri}?error=access_denied&state={state}
```

d) Exchange authorization code for access token

After the resource owner granted access to the client application in step c, the client application can use the temporary authorization code to request an access token.

```
POST /token
```

In this request you should specify an **Authorization** header to authenticate the client application using the HTTP Basic authorization scheme. The header should contain the Client ID and Client Secret strings concatenated with : and base64 encoded. For example:

```
//base64_encode({client_id}:{client_secret});

Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
```

The request may include the following parameters as JSON in the request body:

Parameter	Type	Description
code	string	Required. Authorization code that should be exchanged for an access token.
grant_type	string	Required. Value: "authorization_code"
redirect_uri	string	Required. The callback URL of the client application.
scope	string	Optional. The scope of the access request, represented as a list of space-delimited, case-sensitive strings.

Request body example:

```
{
  "code": "ZdadQ538tzYbY3ab8as7T8asd7sf6c5",
  "grant_type": "authorization_code",
  "redirect_uri": "https://client.example.com/callback"
}
```

As an alternative to the authorization header the Client ID and Secret can also be send in the body using the following parameters:

Parameter	Type	Description
client_id	string	Required. The client ID given for application after registration in Kwebbl.
client_secret	string	Required. The client secret given for application after registration in Kwebbl.

Request body example:

```
{
 "code": "ZdadQ538tzYbY3ab8as7T8asd7sf6c5",
 "grant_type": "authorization_code",
 "redirect_uri": "https://client.example.com/callback"
}
```

Using both the authorization header and adding the Client ID and Secret in the request body is not allowed.

e) Response with access token

The Authorization server will respond with a JSON object that contains the access token, the token type, the lifetime in seconds and a refresh token. The access token is used to access API resources whilst the refresh token is used to request a new access token if the lifetime has been exceeded.

Response example:

```
{
   "access_token": "XnwprKzFTprxzHyQ5aogpb9ao9agmtNd",
   "refresh_token": "06kx3EudiaIjww2hfi6OFNlWXgcUjB3f",
   "expires_in": 900,
   "token_type": "Bearer"
}
```

Important: Your application should store both the access and refresh tokens; without them your user will have to go through the whole authorization process every time he is accessing the API!

Important: Make sure to keep your client secret in a safe place, in case of lost or theft it can only be re-generated and it will replace the previous secret. This will make all previously acquired access and refresh tokens invalid!

If the optional **STATE** parameter was passed, it will be included in the response and should be validated.

In case of an error the Authorization Server may respond with the following codes:

HTTP code	Error code	Description
400	invalid_request	Request is missing a required parameter or it's value is invalid, or includes multiple client credentials
401	invalid_client	Client authentication failed (e.g., unknown client or unsupported authentication method).
401	invalid_grant	Authorization grant is invalid, the refresh token is invalid or expired, or was issued to another client.
401	unauthorized_client	Client cannot use this authorization method
500	server_error	Authorization server encountered an unexpected error

Error response example:

```
{
   'error': 'invalid_grant',
   'error_description': 'Refresh token is invalid'
}
```

If you attempted to authenticate using the Authorization request header, the Authorization server will include a "WWW-Authenticate" response header with the same error information.

Using an Access token to make requests

To gain access to any of the APIs resources the Access Token should be send during every request. This is done by adding it to the Authorization header as Bearer type.

For example, if the access token is “mF_9.B5f-4.1JqM” you would make the request as such:

```
GET /resource HTTP/1.1
Host: server.example.com
Authorization: Bearer mF_9.B5f-4.1JqM
```

Typical errors include:

401 – Invalid Grant
Try to generate a new Access Token using the Refresh Token or if that fails re-authorize the user from scratch.

503 – Access Denied
You require additional permissions to access this resource. Refer to the list of available scopes to find out which permission you need and then re-authorize the user(s) from scratch to obtain these permissions.

Restoring / Renewing an Access Token

After expiration, invalidation or loss of an access token the client application can request a new token using the refresh token retrieved in step e. Note that a refresh token expires after 196 days and the Access token usually expires after 14 days.

To request a new access token the steps are very similar to step d and e.

```
POST /token
```

Like previously you should specify an **AUTHORIZATION** header to authenticate the client application using the HTTP Basic authorization scheme. The header should contain the Client ID and Client Secret strings concatenated and base64 encoded. For example:

```
//base64_encode({client_id}:{client_secret});

Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
```

Alternatively the Client ID and Secret can be included in the JSON body as in step d.

The request may include the following parameters as JSON in the request body:

Parameter	Type	Description
refresh_token	string	Required. Refresh token value, given with access token
grant_type	string	Required. Value: "refresh_token"
redirect_uri	string	Optional. The callback URL of the client application.

The response contains the exact same parameters and error codes as specified earlier in step e.

Important: Any previously acquired access or refresh token will be invalidated right after requesting these new tokens.

Available Scopes

Scopes give access to specific kind of API resources, either owned by the user that grants the permissions or owned by other users in the company. Only Admin users may grant permissions to resources of other users in the company.

The available scopes are:

Scope	**Admin Only**	**Description**
user.info		Read your user information.
company.dialplans		Read information about all dialplans in the company, not including the actions inside the dialplan.
company.users		Read user information for all users in the company.
calls.events	X	Subscribe to all types of call events for all users of the company.
calls.events.personal		Subscribe to all types of call events of yourself.
calls.events.presence		Subscribe to limited presence (BLF) events for all users of the company.
calls.manage	X	Perform actions on calls from any user in the company, actions include hangup, transfer, hold and record.
calls.manage.personal		Perform actions on calls from yourself, actions include hangup, transfer, hold and record.
calls.create	X	Start new calls for all users of the company.
calls.create.personal		Start new calls for yourself.

Access Informational Resources

List of Extensions

Endpoint	/extensions
Method	GET
Scopes	company.dialplans
	company.users

Returns a list of extensions, including all users and dialplans in the company.

```
GET /extensions

200 OK
 [
  {
   "id": "p9wprKzFTprxzHyQ5aogpb9ao9agmtNd",
   "name": "Sales Dept",
   "number": 345,
   "external_number": 31102612345,
   “type”: “dialplan”
  },
  {
   "id": "f6kx3EudiaIjww2hfi6OFNlWXgcUjB3f",
   "name": "John Doe",
   "number": 1000,
   “type”: “user”
  },
  …
 ]
```

In case of **success** the response code is **200 OK**, the body will contain a JSON array with extension objects. Each extension object may contain the following parameters:

Parameter	Type	Pattern	Description
id	string	[a-f0-9]{32}	A 128 bits hex describing the extension
name	string	.*	For type user concatenation of first and last name, for type dialplan the name of the dialplan
number	string	\d{3, 15}	Internal extension number of the user or dialplan. May start with 0.
external_number	string	\d{3, 15}	*Optional*. External number of a dialplan in e164 without +. Only appears for type dialplan. May start with 0.
type	enum	(user|dialplan)	The type of extension, could be user or dialplan.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional*. Error code
message	string	*Optional*. A description of the error code

Known error codes are:

HTTP code	Error code	Description
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes

Retrieve a specific extension

Endpoint	/extensions/{extension_id}
Method	GET
Scopes	company.dialplans
	company.users

Returns information about a specific extension identified by **EXTENSION_ID**. An extension can either be a user or a company dialplan.

Examples:

```
GET /extensions/p9wprKzFTprxzHyQ5aogpb9ao9agmtNd

200 OK
 {
   "id": "p9wprKzFTprxzHyQ5aogpb9ao9agmtNd",
   "name": "Sales Dept",
   "number": 345,
   "external_number": 31102612345,
   “type”: “dialplan”
 }
```

```
GET /extensions/f6kx3EudiaIjww2hfi6OFNlWXgcUjB3f

200 OK
 {
   "id": "f6kx3EudiaIjww2hfi6OFNlWXgcUjB3f",
   "name": "John Doe",
   "number": 1000,
   “type”: “user”
 }
```

In case of **success** the response code is **200 OK**, the body will contain a JSON extension object. The extension object may contain the following parameters:

Parameter	Type	Pattern	Description
id	string	[a-f0-9]{32}	A 128 bits hex describing the extension
name	string	.*	For type user concatenation of first and last name, for type dialplan the name of the dialplan
number	string	\d{3, 15}	Internal extension number of the user or dialplan. May start with 0.
external_number	string	\d{3, 15}	*Optional.* External number of a dialplan in e164 without +. Only appears for type dialplan. May start with 0.
type	enum	(user|dialplan)	The type of extension, could be user or dialplan.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occured
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes
404	entity_not_exist	The extension with the specified ID does not exist

Subscribing to Realtime Events

List of subscriptions

Endpoint	/subscriptions
Method	GET
Scopes	calls.events
	calls.events.personal
	calls.events.presence

Returns a list of all subscriptions.

```
GET /subscriptions

200 OK
[
  {
    "id": "a5dc443c0983ff28edf4a11798fcecfe",
    "events": [
      "created",
      "ringing",
      "answered",
      "terminated"
    ],
    "agents": [
      {
        "field": "*",
        "extension_id": "*",
        "number": "*"
      }
    ],
    "mode": "presence",
    "callback_url": "http://example.com/callback",
    "callback_content_type": “application/json”
  },
  …
]
```

In case of **success** the response code is **200 OK**, the body will contain a JSON array with subscription objects. Each subscription object may contain the following parameters:

Parameter	Type	Pattern	Description
id	string	[a-f0-9]{32}	A 128 bits hex describing the subscription
events	array		*Optional.* A list of events to which you are subscribed. Default is ["created", "ringing", "answered", "terminated"]. Also refer to “Available events”.
agents	array		*Optional.* A list of agents (extensions) to which you are subscribed. Default is [{"number": "*"}].
agents[].field	enum	*|from|to	*Optional.* The field in which to search for the number or extension_id. * means both in from (caller) and to (callee).
agents[].number	string	\d{3, 15}	*Optional.* The internal number of an extension (user or dialplan) you want to subscribe to. * means all
agents[].extension_id	string	[a-f0-9]{32}	*Optional.* A 128 bits hex describing the specific extension you want to subscribe to. * means all
mode	enum	(presence|detailed)	*Optional.* Default is “presence”. Presence will contain limited information about the calls and is used for BLF functionality. Detailed will contain extended information about the call and parties involved.

callback_url	string	.*	Notifications will be POSTed to this URL. This URL should be a full path, including protocol and domain.
callback_content_type	enum	application/json	*Optional*. The content type of the notifications. Currently only application/json is supported.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	**Type**	**Description**
status_code	number	HTTP Status Code
error_code	string	*Optional*. Error code
message	string	*Optional*. A description of the error code

Known error codes are:

HTTP code	**Error code**	**Description**
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes

Retrieve a specific subscription

Endpoint	/subscriptions/{subscription_id}
Method	GET
Scopes	calls.events
	calls.events.personal
	calls.events.presence

Returns a specific subscription identified by **SUBSCRIPTION_ID**.

```
GET /subscriptions/a5dc443c0983ff28edf4a11798fcecfe

200 OK
 {
  "id": "a5dc443c0983ff28edf4a11798fcecfe",
  "event_types": [
    "created",
    "ringing",
    "answered",
    "terminated"
  ],
  "agents": [
    {
      "field": "*",
      "extension_id": "*",
      "number": "*"
    }
  ],
  "mode": "presence",
  "callback_url": "http://example.com/callback",
  "callback_content_type": “application/json”
 }
```

In case of **success** the response code is **200 OK**, the body will contain a JSON array with subscription objects. Each subscription object may contain the following parameters:

Parameter	Type	Pattern	Description
id	string	[a-f0-9]{32}	A 128 bits hex describing the subscription
event_types	array		*Optional.* A list of events to which you are subscribed. Default is ["created", "ringing", "answered", "terminated"]. Also refer to "Available events".
agents	array		*Optional.* A list of agents (extensions) to which you are subscribed. Default is [{"number": "*"}].
agents[].field	enum	*|from|to	*Optional.* The field in which to search for the number or extension_id. * means both in from (caller) and to (callee).
agents[].number	string	\d{3, 15}	*Optional.* The internal number of an extension (user or dialplan) you want to subscribe to. * means all
agents[].extension_id	string	[a-f0-9]{32}	*Optional.* A 128 bits hex describing the specific extension you want to subscribe to. * means all
mode	enum	(presence|detailed)	*Optional.* Default is "presence". Presence will contain limited information about the calls and is used for BLF functionality. Detailed will contain extended information about the call and parties involved.
callback_url	string	.*	Notifications will be POSTed to this URL. This URL should be a full path, including protocol and domain.

callback_content_type	enum	application/json	*Optional*. The content type of the notifications. Currently only application/json is supported.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional*. Error code
message	string	*Optional*. A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes
404	entity_not_exist	The subscription with the specified ID does not exist

Request a new subscription

Endpoint	/subscriptions
Method	POST
Scopes	calls.events
	calls.events.personal
	calls.events.presence

Requests a new subscription according to the settings provided in the request body. Settings include filters to specify which types and for which agents you would like to receive.

```
POST /subscriptions
{
  "event_types": [ "*" ],
  "mode": "detailed",
  "callback_url": "http://example.com/callback"
}
```

The available parameters are:

Parameter	Type	Pattern	Description
id	string	[a-f0-9]{32}	A 128 bits hex describing the subscription
event_types	array		*Optional*. A list of events to which you are subscribed. Default is ["created", "ringing", "answered", "terminated"]. Also refer to "Available events".
agents	array		*Optional*. A list of agents (extensions) to which you are subscribed. Default is [{"number": "*"}].
agents[].field	enum	*|from|to	*Optional*. The field in which to search for the number or extension_id. * means both

			in from (caller) and to (callee).
agents[].number	string	\d{3, 15}	*Optional*. The internal number of an extension (user or dialplan) you want to subscribe to. * means all
agents[].extension_id	string	[a-f0-9]{32}	*Optional*. A 128 bits hex describing the specific extension you want to subscribe to. * means all
mode	enum	(presence|detailed)	*Optional*. Default is "presence". Presence will contain limited information about the calls and is used for BLF functionality. Detailed will contain extended information about the call and parties involved.
callback_url	string	.*	Notifications will be POSTed to this URL. This URL should be a full path, including protocol and domain.
callback_content_type	enum	application/json	*Optional*. The content type of the notifications. Currently only application/json is supported.

Notes about agents[] fields:

- If agents[].field is not empty, but agents[].extension_id and agents[].number are empty than agents[].field will be changed to "*";
- If agents[].extension_id and agents[].number are both provided, events will be filtered according to agents[].extension_id and agents[].number will be changed to "*".

Notes about scopes and permissions:

- calls.events – No limitations to filters in agents and events field.
- calls.events.personal – Filters in agents only allow receiving events involving the user that requested the used access token. No limitations on events field.

- calls.events.presence – Only allows events "created", "ringing", "answered", "terminated" and mode "presence". No limitations on agents field.

Examples
Subscribe to all events for all users in the company. *(note: call.events scope required)*

```
{
 "event_types": [ "*" ],
 "mode": "detailed",
 "callback_url": "http://example.com/callback"
}
```

Minimal settings, subscribe to presence events of all users in the company.

```
{
 "callback_url": "http://example.com/callback"
}
```

Subscribe to presence events, only receive the start and stop of the call and only if user 001 is the caller.

```
{
 "event_types": [
  "created",
  "terminated"
 ],
 "mode": "presence",
 "agents": [
  {
   "field": "from",
   "number": "001"
  }
 ],
 "callback_url": "http://example.com/callback",
}
```

In case of **success** the response code is **200 OK**, **201 Created** or **202 Accepted**. The body contains the just created subscription.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description

status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes

Cancel a subscription

Endpoint	/subscriptions/{subscription_id}
Method	DELETE
Scopes	calls.events
	calls.events.personal
	calls.events.presence

Cancel a specific subscription identified by **SUBSCRIPTION_ID**.

```
DELETE /subscriptions/a5dc443c0983ff28edf4a11798fcecfe

200 OK

{
  "ok": true
}
```

In case of **success** the response code is **200 OK**.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes
404	entity_not_exist	The subscription with the specified ID does not exist

Modify a specific subscription

It is not possibly to modify a specific subscription. To modify just cancel the current subscription and then create new one.

Receiving events from subscriptions

Event Structure

After subscribing to events you will start receiving POST requests on the supplied **CALLBACK_URL**. The base structure of these events looks as following:

Parameter	Type	Pattern	Description
id	string	.*	A unique identifier of the event
call_id	string	[a-f0-9]{32}	A unique identifier of the call (leg)
event_type	enum		One of the event types as described in "Available events".
time	string	.*	Date and Time of event according to ISO 8601. Includes milliseconds and usually is UTC.
from	object		An object describing the agent that is the caller.
via	object		An object describing an agent that is involved in forwarding the call between {from} and {to}. If included this usually describes a dialplan.
to	object		An object describing the agent that is the callee.
direction	enum	inbound|outbound	Direction of the call.

The agent object in from and to has at least one of these properties:

Parameter	Type	Pattern	Description
number	string	\+?\d{3,15}	The internal number of an extension (user or dialplan), the external number of a dialplan or the external number of an external destination. External numbers are in e164 format and may or may not be prefixed with a + or 00.
extension_id	string	[a-f0-9]{32}	A 128 bits hex describing a specific extension (user or dialplan)
caller_id	string	.*	The CallerID as used by the agent, could anonymous.

Note: Transfered and Bridged events differ from this structure

Available events

- created – The call (leg) was created
- ringing – The call (leg) is ringing
- answered – The call (leg) was answered
- terminated – End of the call (leg)
- transferred – The call (leg) was transferred
- bridged – 2 call legs were bridged together
- holding – The call (leg) is on hold
- resume – The call (leg) resumed from hold
- start_recording – Call recording was started for the call (leg)
- stop_recording – Call recording was stopped for the call (leg)

Note: As events are received and send asynchronously and distributed, the order of events could change from call to call.

Created

```
{
  "id": "a1b2c3d0",
  "event_type": "created",
  "from": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "caller_id": "anonymous"
  },
  "to": {
    "number": "31101111111"
  },
  "direction": "inbound",
  "time": "2016-08-30T14:02:06.435Z"
}
```

Ringing

```
{
 "id": "a1b2c3d2",
 "event_type": "ringing",
 "call_id": "4a2bcad2-c4b9e70a",
 "from": {
  "extension_id": "abcdef123456abcdef123456abcdef12",
  "caller_id": "anonymous"
 },
 "to": {
  "number": "31101111111"
 },
 "direction": "inbound",
 "time": "2016-08-30T14:02:06.635Z"
}
```

Answered

```
{
 "id": "a1b2c3d3",
 "event_type": "answered",
 "call_id": "4a2bcad2-c4b9e70a",
 "from": {
  "extension_id": "abcdef123456abcdef123456abcdef12",
  "caller_id": "anonymous"
 },
 "to": {
  "number": "31101111111"
 },
 "direction": "inbound",
 "time": "2016-08-30T14:02:16.635Z"
}
```

Bridged

Parameters **CALL_ID**, **FROM**, **TO** and **DIRECTION** are moved to separate objects for each respective call leg inside of the **CALLS** parameter.

```
{
 "id": "a1b2c3d3",
 "event_type": "bridged",
 "calls": [
  {
   "call_id": "4a2bcad2-c4b9e70a",
   "from": {
    "number": "31101111111"
   },
   "to": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "number": "31101222222"
   },
   "direction": "inbound"
  },
  {
   "call_id": "fd8265472-8c34a51b",
   "from": {
    "number": "31101111111"
   },
   "to": {
    "extension_id": "def456789abcdef456789abcdef45678",
    "number": "030"
   },
   "direction": "outbound"
  }
 ]
 "time": "2016-08-30T14:02:16.635Z"
}
```

Holding

```
{
  "id": "a1b2c3d4",
  "event_type": "holding",
  "call_id": "4a2bcad2-c4b9e70a",
  "from": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "caller_id": "anonymous"
  },
  "to": {
    "number": "31101111111"
  },
  "direction": "outbound",
  "time": "2016-08-30T14:03:06.635Z"
}
```

Resumed

```
{
  "id": "a1b2c3d5",
  "event_type": "resumed",
  "call_id": "4a2bcad2-c4b9e70a",
  "from": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "caller_id": "anonymous"
  },
  "to": {
    "number": "31101111111"
  },
  "direction": "outbound",
  "time": "2016-08-30T14:03:36.635Z"
}
```

Start Recording

```
{
  "id": "a1b2c3d6",
  "event_type": "start-recording",
  "call_id": "4a2bcad2-c4b9e70a",
  "from": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "caller_id": "anonymous"
  },
  "to": {
    "number": "31101111111"
  },
  "direction": "outbound",
  "time": "2016-08-30T14:05:36.635Z"
}
```

Stop Recording

```
{
  "id": "a1b2c3d7",
  "event_type": "stop-recording",
  "call_id": "4a2bcad2-c4b9e70a",
  "from": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "caller_id": "anonymous"
  },
  "to": {
    "number": "31101111111"
  },
  "direction": "outbound",
  "time": "2016-08-30T14:15:36.635Z"
}
```

Terminated

```
{
  "id": "a1b2c3d8",
  "event_type": "terminated",
  "call_id": "4a2bcad2-c4b9e70a",
  "from": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "caller_id": "anonymous"
  },
  "to": {
    "number": "31101111111"
  },
  "direction": "outbound",
  "reason": "hangup"
  "time": "2016-08-30T14:20:36.635Z"
}
```

Transferred

Parameters **FROM** and **TO** are moved to **BEFORE** and **AFTER** to reflect the state before and after the transfer was made. The **INITIATOR** contains information about the agent that initiated the transfer, whilst the optional field **TRANSFER_TYPE** shows if the transfer was done using blind or attended transfer.

```
{
  "id": "a1b2c3d9",
  "event_type": "transferred",
  "call_id": "4a2bcad2-c4b9e70a",
  "before": {
    "from": {
      "extension_id": "abcdef123456abcdef123456abcdef12",
      "caller_id": "anonymous",
      "number": "010"
    },
    "to": {
      "number": "31101111111"
    }
  },
  "after": {
    "from": {
      "extension_id": "def456789abcdef456789abcdef45678",
      "caller_id": "anonymous",
      "number": "030"
    },
    "to": {
      "number": "31101111111"
    }
  },
  "initiator": {
    "extension_id": "abcdef123456abcdef123456abcdef12",
    "number": "020"
  },
  "transfer_type": "blind",
  "time": "2016-08-30T14:20:36.635Z"
}
```

Perform Realtime Call Actions

Placing a new call

Endpoint	/calls
Method	POST
Scopes	calls.create
	calls.create.personal

Place a new call between 2 agents, agents can be external numbers, users and dialplans.

```
POST /calls

{
 "from": {
  "number": "010"
 },
 "to": {
  "number": "+31203080700"
 }
}

202 Accepted
```

Limitations

- Creating calls between 2 external numbers is not allowed as part of anti-fraud measures
- The scope calls.create.personal only allows making calls from the resource owner/user of the access token
- From dialplan to external is reversed and will always act as external to dialplan
- From dialplan to user is reversed and will always act as user to dialplan

The parameters below are accepted in the JSON body of the request:

Parameter	Type	Pattern	Description
from	object		*Optional.* From which agent should the call originate. Default will be the user/owner of the access token.
from.number	string		The internal number of an extension or dialplan, the external number of a dialplan or the external number of an external destination.
from.extension_Id	string	[a-f0-9]{32}	A 128 bits hex describing the extension. (user or dialplan)
to	object		To which agent should the call be placed. Either number or extension_id should be set.
to.number	string		The internal number of an extension or dialplan, the external number of a dialplan or the external number of an external destination.
to.extension_Id	string	[a-f0-9]{32}	A 128 bits hex describing the extension. (user or dialplan)

In case of **success** the response code is **202 Accepted**.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes

Manipulating an existing call

Terminate / Hangup

Endpoint	/calls/{call_id}/terminate /calls/{call_id}/hangup
Method	POST
Scopes	calls.manage
	calls.manage.personal

Terminate / Hangup a call identified by **CALL_ID**.

```
POST /calls/4a2bcad2-c4b9e70a/terminate

202 Accepted
```

In case of **success** the response code is **202 Accepted**.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes
404	entity_not_exist	A call with that Call ID does not exist

Transfer

Endpoint	/calls/{call_id}/transfer
Method	POST
Scopes	calls.manage
	calls.manage.personal

Transfer blind or attended a call identified by **CALL_ID**. To finish attended transfer use * on your phone, to cancel attended transfer use # on your phone.

```
POST /calls/4a2bcad2-c4b9e70a/transfer
{
  "target": {
    "number": "31101111111"
  },
  "transfer_type": "attended"
}

202 Accepted
```

```
POST /calls/4a2bcad2-c4b9e70a/transfer

{
  "target": {
    "extension_id": "abcd1234abcd1234abcd1234abcd1234"
  },
  "transfer_type": "blind"
}

202 Accepted
```

```
{
  "target": {
    "number": "030"
  }
}

202 Accepted
```

Accepted parameters are:

Parameter	Type	Pattern	Description
target	object		Where to transfer the other person. Either number or extension_id should be set.
target.number	string		The internal number of an extension or dialplan, the external number of a dialplan or the external number of an external destination.
target.extension_Id	string	[a-f0-9]{32}	A 128 bits hex describing the extension. (user or dialplan)
transfer_type	enum	blind|attended	What kind of transfer to perform. Default is blind.

In case of **success** the response code is **202 Accepted**.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes
404	entity_not_exist	A call with that Call ID does not exist

Hold and Resume

Endpoint	/calls/{call_id}/hold /calls/{call_id}/resume
Method	POST
Scopes	calls.manage
	calls.manage.personal

Hold and resume a call identified by **CALL_ID**.

```
POST /calls/4a2bcad2-c4b9e70a/hold

202 Accepted

POST /calls/4a2bcad2-c4b9e70a/resume

202 Accepted
```

In case of **success** the response code is **202 Accepted**.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes
404	entity_not_exist	A call with that Call ID does not exist

Call Recording

Endpoint	/calls/{call_id}/start-recording /calls/{call_id}/stop-recording
Method	POST
Scopes	calls.manage
	calls.manage.personal

Start and stop recording for a call identified by **CALL_ID**.

```
POST /calls/4a2bcad2-c4b9e70a/start-recording

202 Accepted

POST /calls/4a2bcad2-c4b9e70a/stop-recording

202 Accepted
```

In case of **success** the response code is **202 Accepted**.

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
400	validation	A validation error on the input occurred
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes
404	entity_not_exist	A call with that Call ID does not exist

Multiple CallerID’s

Get list of CallerID’s

Endpoint	/extensions/{extension_id}/multiple_caller_ids
Method	GET
Scopes	company.dialplans
	company.users

Get list of available CallerID’s for an extension

GET /extensions/3338d8e9db15becc3397a47500dac7e0/multiple_caller_ids

```
200 OK
[
    { caller_id: "2678d8e9db15becc3397a47500dac7e0", caller_number: "31012345678"},
    { caller_id: "2678d8e9db15becc3397a47500dac543", caller_number: "31012345679"}
]
```

In case of **success** the response code is **200 OK**, the body will contain a JSON array with CallerID objects. Each CallerID object may contain the following parameters:

Parameter	Type	Pattern	Description
caller_id	string	[a-f0-9]{32}	A 128 bits hex describing the extension
caller_number	string	\d{3, 15}	External number in e164 without +. Available for user to set as CallerID

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional*. Error code
message	string	*Optional*. A description of the error code

Known error codes are:

HTTP code	Error code	Description
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes

Change user CallerID

Endpoint	/extensions/{extension_id}/caller_id
Method	PUT
Scopes	company.dialplans
	company.users

Updates user CallerID.
Send PUT request with JSON object in the body indicating id of the number you wish to set as user CallerID.
User with Admin role can change CallerID of anyone in the company.

PUT /extensions/3338d8e9db15becc3397a47500dac7e0/caller_id

```
{
    "caller_id": "2678d8e9db15becc3397a47500dac7e0"
}
```

```
200 OK
{
    "extension_id": "3338d8e9db15becc3397a47500dac7e0",
    "caller_id": "2678d8e9db15becc3397a47500dac7e0"
}
```

Accepted parameters are:

Parameter	Type	Pattern	Description
caller_id	string	[a-f0-9]{32}	Required. A 128 bits hex id of one of the available numbers

In case of **success** the response code is **200 OK**, the body will contain a JSON object with extension id and CallerID. Each CallerID object may contain the following parameters:

Parameter	Type	Pattern	Description
extension_id	string	[a-f0-9]{32}	A 128 bits hex describing the extension
caller_id	string	[a-f0-9]{32}	A 128 bits hex id of the number used as CallerID

In case of an **error**, the response code is anything other than 200 OK. The body may contain a JSON object describing the error:

Parameter	Type	Description
status_code	number	HTTP Status Code
error_code	string	*Optional.* Error code
message	string	*Optional.* A description of the error code

Known error codes are:

HTTP code	Error code	Description
401	invalid_grant	Authorization grant is invalid or expired
403	access_denied	No permission to access the required scopes

Changelist

Version	Chapter	Description
1.0.0 - Rev 16	Authorization Code Grant Flow -> - d) Exchange authorization code for access token - Restoring / Renewing an Access Token	Changed format of authorization header.
1.0.0 - Rev 16	Subscribing to Realtime Events -> - Retrieve a specific subscription - Request a new subscription	Changed parameter name **EVENTS** to **EVENT_TYPES**
1.0.0 - Rev 16	Receiving events from subscriptions	Added parameter **VIA** to events.
1.0.0 – Rev 17	Multiple CallerID	Added the new Multiple CallerID information