PURECLOUD

together with **CS Comms**

Istra 9.2 REST API

Sign In On Behalf of another Administrator

```
GET /restletrouter/v1/logs/Voicemail/ HTTP/1.1
Authorization: Basic V2hhdCBkaWQgeW91IGV4cGVjdD8gOy0p
Host: restletrouter.supertelephony.net
Accept: application/json
X-Application: myTelephony;17e09411-d650-4801-98db-
0200660ac681
Accept-encoding: gzip,deflate
```

Pure Cloud Solutions Ltd.
www.purecloudsolutions.co.uk

6 The Pavillions, Amber Close
Tamworth, B77 4RP

T: 0333 150 6780

I. Table of contents

I. Table of contents 2
II. Non-disclosure agreement 3
III. Trademarks 3
V. About this document 3
VI. Introduction 4
A. Typical use case: integrating myTelephony within an Operator's extranet 4
B. API goals 5
VII. API Security 6
A. Administrator accounts 6
B. HTTPS only 7
C. Defenses against brute force attacks 7
VIII. API usage requirements 8
A. Dedicated 'Source" Administrator technical account 8
B. Automatic and consistent provisionning of "Target" Administrator accounts 8
C. Override of the myTelephony login page URI 9
IX. API general usage 10
A. HTTP Host 10
B. Mandatory HTTP header: Content-Type 10
C. Recommended HTTP header: Accept-Encoding 11
D. Error handling 11
X. Sign In On Behalf of Another Administrator 12
A. General principle 12
B. REST Resource: /transferLogin 13
XI. Common issues 14

II. Non-disclosure agreement

All information included in this document is the entire property of Pure Cloud Solutions and as such, must stay confidential.

Access to this document is restricted to those companies or parties having signed a Non-Disclosure Agreement (NDA) with Pure Cloud Solutions.

Diffusing information to other parties without a signed Non-Disclosure Agreement between CPure Cloud Solutions and the other party is strictly forbidden.

III. Trademarks

Centile™ and Istra™ are trademarks of Centile Telecom Applications SAS.

V. About this document

The goal of this document is to present the REST API method to "Sign In On Behalf of Another Administrator", and is intended for an audience of developers.

The document considers you are already familiar with the REST API over HTTP principles[1], and the JSON format[2].

Also, because of the ubiquity of the HTTP protocol, the examples given in this document are not tied to a particular programming language: instead, the command line tool *curl*[3] is preferred, since it is available for the three major operating systems, in addition to offer a convenient abstraction for the sake of readability.

1 https://en.wikipedia.org/wiki/Representational_state_transfer 2 https://fr.wikipedia.org/wiki/JavaScript_Object_Notation
3 http://curl.haxx.se/

VI. Introduction

Before delving into the details of the API, a few words about the typical use case where it applies, and then what are its goals.

Please note this API has meen made available starting from Istra 9.2.1.5 release and beyond.

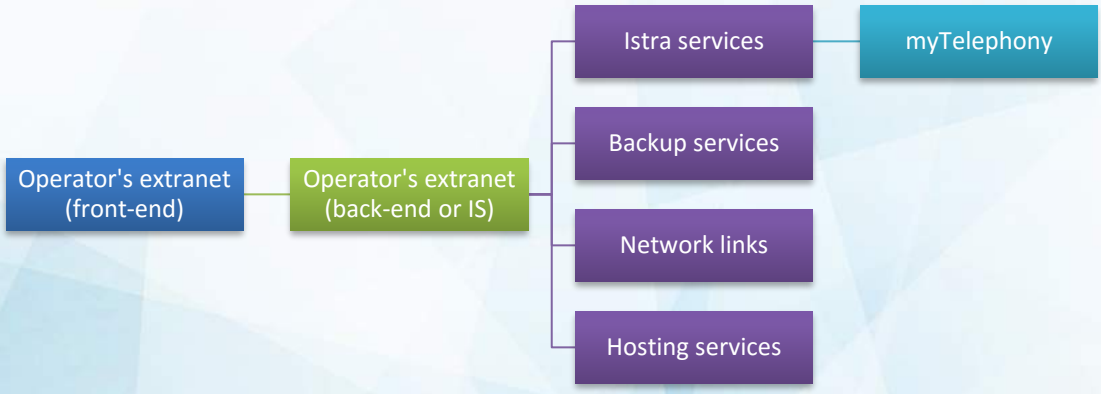
A. Typical use case: integrating myTelephony within an Operator's extranet

We explain here a typical use case for which the API makes sense.

Say an Operator (O) has an Istra platform: O may offer other services as well (hosting or backups services, network links, …) managed by O's Information System (IS). Most probably, this IS will provide an extranet (E) access for O's customers, in order for them to manage their accounts (e.g.: for the sake of self-care, billing, ordering, etc…).

A typical integration use case is where a customer C signs in the extranet, and clicks on a "go to myTelephony" in order to land in myTelephony: however, for the sake of the user experience, this user should not be presented the myTelephony's login page, but directly the main landing page instead.

A general architecture looks like the following:



Such an integration can be done using two approaches:

1. There is a SSO mechanism set up, onto which Istra delegate its authentication request to. This is applicable under several conditions:
    a. there is a standardized central authentication service (e.g.: Kerberos with Microsoft Active Directory, LDAP, etc…)
    b. the Istra platform has a direct network connectivity to this central authentication
    c. all customers' identifiers exist in both this central authority and Istra, which requires users provisioning and synchronization in Istra
2. Another simpler but effective solution is to use the API presented in this document, in order to use a special account dedicated to the extranet usage, which will be able to create myTelephony session on behalf of any extranet users. Conditions to meet are:
    a. all customers' identifiers exist in both the extranet and Istra realms

b. a dedicated technical Administrator account is created in Istra, and uses the "Sign In On Behalf of Another Administrator" API in order to create the myTelephony's session on behalf of the extranet's users

In general, the first solution is preferred but it is often applicable to large deployment only where all Istra subscribers are managed by the same central authentication service or in the same Istra enterprise (as opposed to multiple different enterprise, managed by different authentication services – when such services exist)

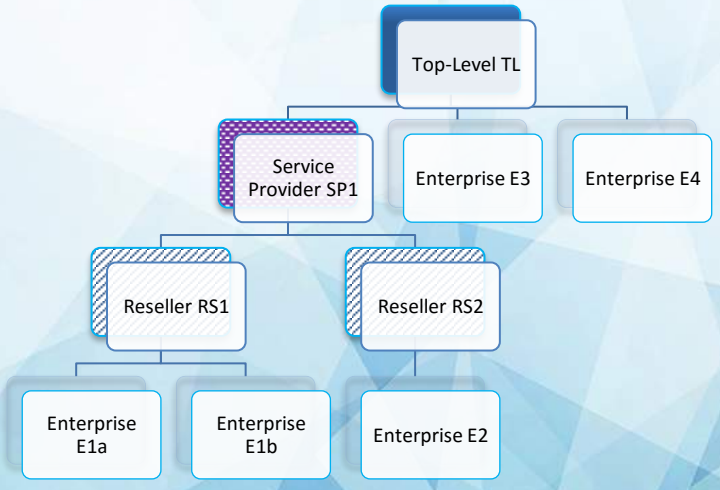
When no standard SSO mechanism is in place, the second option listed above offers an interesting alternative that preserve both security and integration cost integration, while keeping the interactions between the systems simple and easy to implement.

B. API goals

This API has been designed with 2 goals in mind:

1. For a given dedicated Administrator account (i.e.: the origin account) who is able to sign in to the REST API, make it possible to "create" and "transfer" a myTelephony session into another Administrator's account session (i.e.: the target account), without the need of the target account's credentials[4]
2. Respect the Istra administrative domain hierarchy: an origin account can sign in on behalf of a another target account only when the target account is under its administrative scope.

This last point above is depicted below, where an Istra platform has the following administrative domains tree:



In figure above, sign in on behalf would be permitted from these Administrator's source account to this Administrator target accounts:

[4] As an analogy, one may compare it to the UNIX `su` command, where it is similarly used by a computer user to execute commands or starts another session with the privileges of another user account, provided the UNIX permissions are followed

Source accounts	Authorized target accounts
TL	All
SP1	RS1, RS2, E1a, E1b, E2
RS1	E1a or E1b
RS2	E2

Conversely, the following source accounts cannot sign in on behalf of these target accounts:

Source accounts	Unauthorized target accounts
E1a, E1b, E2, E3, E4	All
SP1	TL, E3, E4
RS1	TL, SP1, E3, E4, RS2, E2
RS2	All but E2

VII. API Security

This paragraph covers the main security aspects about the API.

A. Administrator accounts

First, only Administrators accounts can use this API, thanks to their login and password. Some details about them:

- Logins:
  - Are created / chosen by the administrator of the platform, and cannot be changed by their corresponding user.
  - They could be an arbitrary text string (e.g. “jsmith”), unique on the platform among all Administrators accounts (emails are natural good candidates, since they are unique by nature - e.g.: jsmith@supertelephony.net - , but are not enforced)
  - Allowed characters are only the following: `0-9a-zA-Z\-_\.!~*'()%@`
- Passwords:
  - Are arbitrary strings of text, among the allowed characters `0-9a-zA-Z\-_\.!~*'()%@`
  - Must comply with the **Administrator Password Policy**[5] set up on the platform.
  - Such passwords are stored in a secured manner, as non reversible hashes, salted first with a dynamic salt, and a second time with a static one. They cannot reasonably be reverse engineered. They are ever transmitted in clear by Istra, and there is no possibility to recover them, only to change them.

[5] **C.f. webAdmin > Adminitrative Domains > Top-Level > edition screen**

Regarding passwords, only the technical Administrator account needs a password: indeed the target Administrator account may have a random password never known by anyone since all sign in requests would be made through the API by the technical account.

For the remaining of this document, examples given will assume the login `myOriginLogin` and the password `myOriginPassword` for the origin account of the technical Administrator account that will use the API, and `myTargetLogin` for the customer's Administrator target account.

B. HTTPS only

Istra is made from scratch to run only on HTTPs (HTTP over TLS), and this applies to this REST API.

This particularly means you must have a valid SSL certificate that is trusted by the client end point.

If you don't use a valid SSL certificate, you still have the option to ignore the server certificate during the SSL handshake: while this is convenient for development or test purpose, Centile does not recommend in any way to rely on such weak SSL environment in production.

That being clear, please note the `curl` tool can ignore the invalid server certificate thanks to its option `--insecure`. As a result, everywhere a `curl` example appears in this document, you could optionally add this option to run the example in an invalid SSL environment. And most probably, your preferred programming language has a way to deal similarly (or, alternatively, you would find a way to force to trust your invalid certificate: details vary according to your programming environment).

Alternatively, you still have the option to reach the Istra REST API end point over plain HTTP, but while this can be made, securoty constraints apply : for instane be sure the network is trusted from end to end, and that there is no risk to expose this plan text traffic.

C. Defenses against brute force attacks

One might fear Administrators accounts could be compromise, especially if the REST API is exposed to Internet for instance and subject to brute force attacks

It is not necessary mandatory to expose the REST API over Internet: it all depends of your system & network architecture and/or needs: perhaps only a back-end access is enough for instance.

However, in case the API is exposed on Internet, please know that Istra supports a Password Policy (c.f. first paragraph above) that helps you defeat malicious usage:

- It can define acceptable passwords (length, complexity…)
- Also, accounts can automatically be suspended in case of too many failures in a row: you can configure the associated
- Sessions are time limited, configurable on the platform
- Be a database dump leaked, it will be very difficult for an attacker – if practical – to reverse engineer any Administrators password

More about security is described in another document[6], availble on demand.

VIII. API usage requirements

A. Dedicated 'Source" Administrator technical account

It is preferable to create a specific Administrator account in Istra, that will be in charge of the "Sign In On Behalf" requests in the operator's extranet, for separation of duties reasons. Its login and password will be used by the operator's extranet, and it is recommended it is used only for that.

As a reminder, and for the reasons exposed in previous paragraph § VI.B. "API goals", this administrator account must have an administrative domain where the target Administrator account is in its visible administrative scope (could be any of applicable Top Level, Service Provider, or Reseller type of Administrator account), otherwise it will refused the Sign In On Behalf requests.

Also, because of the principle of least privilege, its rights must be reduced in order to forbid access to others administrative user interface (webAdmin, myReports, ACD Stats, CDR Browser and myRecordings, and all future web application as well). This is required in order to make the account a strictly technical account, usable only for API calls purpose, and nothing else.

The only right access it needs is the myTelephony one, to be able to create session for this web application. In Istra, one would see in webAdmin > Administrators > (your given Administrator account) > edition screen the following settings to use:

fields subject to license	
Web Administration access	(•) no ( ) yes
myTelephony access	( ) no (•) yes
myReports access	(•) no ( ) yes
ACD Stats access	(•) no ( ) yes
CDR Browser access	(•) no ( ) yes
Call Recording browser access	(•) no ( ) yes
log on behalf of end users	(•) no ( ) yes

Figure 1 Example of a technical Administrator settings in webAdmin (menu Rights > Administrators > edition)

Please note in the figure above that the right "log on behalf of end users" does not need to be granted: indeed, it is required in order to create sessions for end users accounts (read: Istra telephony subscribers), but it is not relevant to Administrators account. Indeed, regarding Administrators, the right is implicit because of the multi-tier / multi-tenant built-in aspect of Istra.

Of course, this Administrator login and especially its password must be kept private; the best security practices about password usage in production systems apply here, and the Operator's IS may have specific rules to comply with as well.

B. Automatic and consistent provisionning of "Target" Administrator accounts

[6] **Namely "Istra Security Information".**

For the API to be used, the target customer's Administrator accounts must already exist in Istra prior to the Sign In On Behalf requests by the source account; otherwise it will simply fail.

This is the responsibility of the Operator's Information System to maintain a consistency between the accounts existing in the extranets and in Istra, most notably:

- At creation time, when a new customer is added in the extranet:
  - An Istra Administrator account must be created, with the myTelephony access granted
  - Its identifier must comply with the Istra allowed characters
  - Note about the password for this account:
    - It can be totally random, because it will never be used directly by the target account : all his myTelephony access will be subject to sign in the extranet first.
    - While not mandatory, the IS can try to replicate the user password at creation / update, according to the integration scenario.
- At update time, when an existing customer identifier is changed in the extranet:
  - Apply the same update onto the corresponding Istra Administrator
  - Its identifier must comply with the Istra allowed characters
- At update time, when an existing customer looses or gain the myTelephony access right:
  - Apply the same update onto the corresponding Istra Administrator
- At deletion time, when a customer account is deleted from the extranet:
  - Delete it as well the associated Istra Administrator account

For the creation of Istra Administrators account, please refer to the Istra SOAP Web Services documentation named "Enterprise Provisioning Web Services Developer Guide" [7] that the information System will have to deal with.

C. Override of the myTelephony login page URI

myTelephony uses inactivity expiration timer in order to logout automatically its Administrators (the timer configurable in webAdmin). Any activity in myTelephony after the session expired naturally leads the Administrator to the myTelephony login page, in order to ask again his credentials.

But in such an extranet integration use case, an option for myTelephony is to never display its login page, because:

- the customer will not know which credentials to use in Istra (case of a random password)
- or the customer will try to use its extranet credentials (but this is subject to password replication in extranet and Istra realms)

According to the integration strategy, an option is to override the login page of myTelephony in order to make myTelephony redirect the customer, at session expiration, to the extranet URI.

This is configurable by the Top-Level in Istra webAdmin > Advanced Tools > Cluster configuration > Web Server:

[7] **In particular, look for the functions** `createAdministrator`, `updateAdministrator` **and** `deleteAdministrator`

1. Here, the Top-Level locates and edits the web server that hosts the REST API, whose technical name is `restletrouter`
2. In the subsequent edition screen, Top Level locates the field `command line: advanced options`, and adds the override parameter `-Dmytelephony.landingpage=https://extranet.mydomain.com` (`https://extranet.mydomain.com` being the extranet URI to redirect the customer to)

IX. API general usage

We cover here basic HTTP considerations below, as well as error handling and the common HTTP methods used.

A. HTTP Host

The REST API is exposed onto a host that could take the following appearance:

1. Either `https://restletrouter.mydomain.com/`
2. Or, alternatively, `https://webadmin.mydomain.com/restletrouter/` (where the fragment `webAdmin` can be replaced with `mytelephony` or `myistra` as well, provided they are deployed on the platform).

The important fragments are:

- `restletrouter`, which is actually the API end point name itself
- `mydomain.com`, which is your host name on which the restletrouter is reachable (note: it could be an IP as well), be it private or public according to your topology.

Using one or the other form above just depends of your deployment architecture, and we can assist you determining the best option for your case.

If you plan to use this API to connect Administrators account to myTelephony, please note it is <u>not</u> a requirement to use the same end point that the one used by myTelephony itself: in particular, while a public myTelephony web app uses its own restletrouter (say `https://mytelephony.mydomain.com/restletrouter/`), another restletrouter end point can be used (say `https://my-private-host.local/restletrouter/`) because session meta data are persisted in the Istra database.

The remaining document will assume the generic form `https://HOST/restletrouter/`, and you will have to adapt it to your specific environment.

B. Mandatory HTTP header: Content-Type

When using the API, the `Content-Type` header is required, and specifies the representation to use in requests/responses: so far, only JSON is supported. With `curl`, it translates to the following option (fragment only):

```
curl -H "Content-Type: application/json"
```

C. Recommended HTTP header: Accept-Encoding

The header `Accept-Encoding` is recommended, as part of best practices when it comes to HTTP traffic. Its goam is to compress the HTTP payload in order to save bandwidth. This comes as a little cost, since the compression must be done server side, and decompression at client side, but it could offer some benefits, according to your traffic nature.

Speaking of `curl`, it can be used jointly with the `gunzip` [8]utility on the command line in order to decode zipped server's responses. For instance as follows (fragment only):

```
curl -H "Accept-encoding: gzip,deflate" https://HOST/restletrouter/REST_RESOURCE | gunzip
-c
```

For the sake of readability, examples will not include this header, but we recommend using it in your production environment.

D. Error handling

In all cases, the HTTP status code is semantically valid, and ust be trusted for error hanlidng. Then, in case of:

1. Failed call:
    a. you will — always — obtain an appropriate HTTP status code[9] (`4xx`, `5xx`…)
    b. you will — optionnally — obtain a JSON[10] error response, that details the reason of the error, made of this two properties `code` and `message`:

```
{
"code": "ERR XYZ",
"message": "Some human readable error message detailing the XYZ code."
}
```

    c. (NB: a JSON error message may not be provided in all cases of trivial failures like 401 or 403, where the HTTP status code is enough)
2. Successful call (i.e.: no error occurred during the request processing):
    a. you will — always — obtain a `HTTP 200 OK` (or alternatively a `2xx`)
    b. you will — never — get an error JSON object like the above
    c. you will — always — get the expected content (provided there is one, of course).

[8] Windows users: using the syntax given requires using a Unix type shell because of the pipe (|) usage. You could use for instance http://cmder.net with the full package "msysgit"
[9] https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
[10] https://fr.wikipedia.org/wiki/JavaScript_Object_Notation

X. Sign In On Behalf of Another Administrator

A. General principle

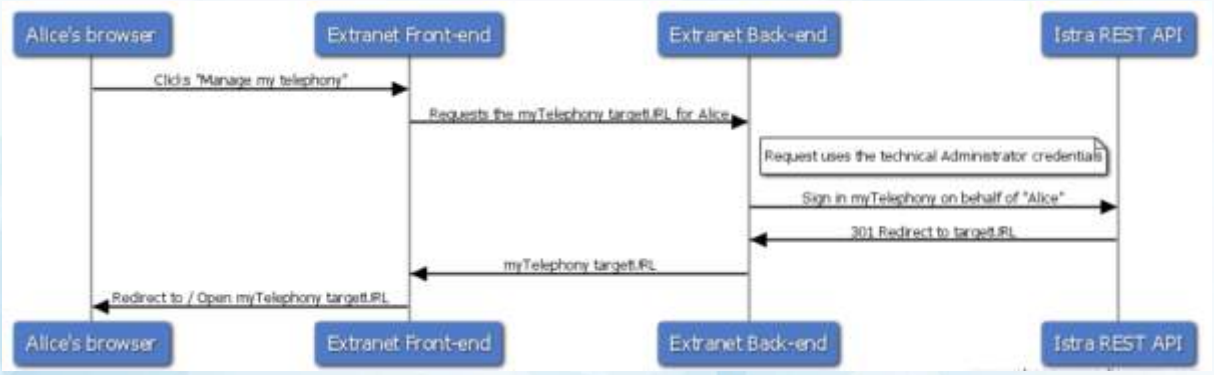
Available only for Administrators, the API uses a simple `HTTP Basic Authentication` with a dedicated syntax that uses the `>` and `:` characters[11] with the three information `myOriginLogin`, `myOriginPassword` and `myTargetLogin` .

In the `curl` world, it translates to the option `-u` to indicate the user credentials (fragment only shown below):

```
curl -u "myOriginLogin>myTargetLogin:myOriginPassword"
```

An expected HTTP answer would then use a status `200 OK` - or more adequately a `301 Moved Permanently` as we'll see later - unless of course a `401 Unauthorized` (missing authentication) or `403 Forbidden` (access is not granted, despite the correct authentication) applies.

Upon success, this creates a session for the target Administrator account, using the origin account's credentials. From here, the general principle for an integration where the user Alice is already signed into the extranet is:



Which can be summarized to:

1. Using the source technical Administrator account, the Extranet requests the Sign In On Behalf of the target Administrator Alice to the Istra REST API
2. Upon API response, the extranet will point Alice to the obtained myTelephony URL (or possibly perform error management if applicable)

Please note that this is a M2M use case : a given Operator's back-end is querying the Istra API, in order to use the response for its own integration usage.

[11] Please note you will have to quote the usage of special characters (like `>`) according to your shell constraints (for instance using a pair of double quote `"` when using the Bash shell, or using the Windows command prompt)

B. REST Resource: /transferLogin

While the principle above does create a session for the target account, we still miss some aspects: we have to ask for which Istra web application the session is to be created, and know how to connect to it once created.

Indeed, the extranet is expected to request the `/transferLogin` REST ressource when using the API with its credentials, with some query strong parameters, which will carry out the two following actions in a single call:

- It creates a new myTelephony session for the target account using the origin credentials
- It then ask to "transfer" this newly created REST session to a resulting myTelephony URL that will be returned in response

The resulting URL contains all the details required in order to point the user's browser to the main page of myTelephony, bypassing the login screen: it takes care of "transferring" all the session identifiers and meta data (session cookie and CSRF security token most notably) to the target application.

A corresponding request with `curl` is:

```
curl -u "myOriginLogin>myTargetLogin:myOriginPassword"
https://HOST/restletrouter/transferLogin?appName=myTelephony&targetAppName=myTelephony
```

Upon success, the API will respond with a `301 Redirect` message, similar to the following (the emphasis shows the most important information to use):

```
HTTP/1.1 301 Moved Permanently
Date: Thu, 23 Mar 2017 08:58:12 GMT
Location: https://HOST/restletrouter/transferLogin?appName=myTelephony&targetAppName=myTel
ephony&sessionID=f4htKLcWLZrJOuUOCtCBzvvVBk%2B4%2FLaU2JsP3rChCoY%3D
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Frame-Options: deny
Content-Security-Policy: default-src 'self'; img-src 'none'
X-Server: IntraSwitch/9.2.31 (Build 1201)
Cache-Control: private
Expires: Thu, 01 Jan 1970 01:00:00 CET
Content-Type: application/json
Content-Length: 0
```

It is then the responsibility of the extranet back-end to check a `HTTP 301` [12] is obtained, and redirect the user to the provided `Location`, using the method of its choice (redirect, open a new tab, open a new window, etc…).

[12] **Please note this `301 Moved Permanently` answer uses precise cache directives in order to not cache this result. This `301` is also meant to be consumed by a back-end (M2M) and not a browser: the requesting back-end could probably use itself safely a `307 Temporary Redirect`, in order to relay the provided `Location` to the customer's browser. We let the back-end's development team make their own sovereign decision for that topic.**

The user can then use myTelephony as expected, with the notable following differences:

1. There is no possibility to change the password in such a myTelephony session because such a change would not update the extranet password
2. At session expiration time, myTelephony will point to the landing page defined in § VIII.C. Override of the myTelephony login page URI, instead of its own login page

Note that when one session is opened in myTelephony through another path than the extranet (i.e.: without Sign In On Behalf), then the session is a regular myTelphony one, in which the Administrator would be able to change his password, which will have no effect on the extranet: as a result, be sure to not let customers open this kind of session.

XI. Common issues

When the API does not behave as expected, check these points:

- Are you using the correct login and password for the orifin account?
- Are you using the correct target login?
- Does the origin Administrator account have the myTelephony right granted in webAdmin > Rights > Administrators > edition screen?
- Same for target Administrator account?
- Is the origin Administrator account blocked for security reason in webAdmin > Rights > Administrators > edition screen?
- Same for target Administrator?
- What is the origin Administrator account type? It can only be a Service Provider or a Reseller type, that has the target Administrator account in its administrative domain scope (a descendant). Alternatively, a Top-Level account will do as well, but this is not recommended for security reasons.
- What is the target Administrator account type? The API is meant to accept an Enterprise type, and it must be visible by the origin account.
- Are you testing against the expected version of Istra, which must be Istra Release >= 9.2.1.6 / Istra build 9.2.29-1289?
- Are you testing against the expected version of myTelephony, which must be Istra Release >= 9.2.1.6 / myTelephony build 9.2.1-62?

Pure Cloud Solutions Ltd.
www.purecloudsolutions.co.uk

6 The Pavillions, Amber Close
Tamworth, B77 4RP

T: 0333 150 6780