



ADSS RAS Developers Guide

ASCERTIA LTD

APRIL 2022

Document Version – 7.0.2

© Ascertia Limited. All rights reserved.

This document contains commercial-in-confidence material. It must not be disclosed to any third party without the written authority of Ascertia Limited.

Commercial-in-Confidence

CONTENTS

| | | |
|----------|---|----------|
| 1 | Introduction | 4 |
| 1.1 | Scope | 4 |
| 1.2 | Intended Readership | 4 |
| 1.3 | Conventions | 4 |
| 1.4 | Technical support..... | 4 |
| 2 | ADSS Server RAS Service Overview | 5 |
| 3 | Business Application Interfaces | 7 |
| 3.1 | Ascertia APIs | 7 |
| 3.1.1 | Service Status | 7 |
| 3.1.2 | Register User | 7 |
| 3.1.3 | Update User | 9 |
| 3.1.4 | Delete User | 11 |
| 3.1.5 | Get User | 12 |
| 3.1.6 | Get Users..... | 13 |
| 3.1.7 | Change Password | 15 |
| 3.1.8 | Recover Password | 16 |
| 3.1.9 | Confirm Recover Password | 18 |
| 3.1.10 | Change User Email..... | 19 |
| 3.1.11 | Confirm Change User Email | 21 |
| 3.1.12 | Change User Mobile | 22 |
| 3.1.13 | Confirm Change User Mobile..... | 24 |
| 3.1.14 | Get Registered Devices..... | 25 |
| 3.1.15 | Delete Device | 26 |
| 3.1.16 | Generate Key Pair..... | 27 |
| 3.1.17 | Delete Key Pair | 29 |
| 3.1.18 | Get CSR | 29 |
| 3.1.19 | Import Certificate | 31 |
| 3.1.20 | Get User's Certificates | 32 |
| 3.1.21 | Authentication/Login without Password..... | 34 |
| 3.2 | CSC APIs..... | 36 |
| 3.2.1 | Authentication/Login..... | 36 |
| 3.2.2 | Authentication/Revoke..... | 38 |
| 3.2.3 | Credentials/List | 39 |
| 3.2.4 | Credentials/Info..... | 40 |
| 3.2.5 | Credentials/Authorize..... | 47 |
| 3.2.6 | Credentials/extendTransaction..... | 50 |

| | | |
|----------|---|-----------|
| 3.2.7 | Credentials/sendOTP | 52 |
| 3.2.8 | Signatures/signHash | 53 |
| 3.2.9 | Application Meta Information..... | 57 |
| 3.2.10 | OAuth2/Authorize | 59 |
| 3.2.11 | OAuth2/Token – Authorization Code Flow..... | 62 |
| 3.2.12 | OAuth2/Token – Client Credentials Flow | 65 |
| 3.2.13 | OAuth2/Token – Refresh Token Flow | 66 |
| 3.2.14 | OAuth2/Revoke..... | 68 |
| 4 | Mobile Application Interfaces | 70 |
| 4.1 | Authenticate Client | 70 |
| 4.2 | Authenticate User | 71 |
| 4.3 | Verify OTPs..... | 73 |
| 4.4 | Renew Access Token..... | 75 |
| 4.5 | Device Registration | 76 |
| 4.6 | List Registered Devices..... | 78 |
| 4.7 | Delete Device..... | 79 |
| 4.8 | Get Pending Authorisation Request..... | 79 |
| 4.9 | Authorise a Pending Request | 81 |
| 4.10 | Cancel a Pending Authorisation Request..... | 83 |
| 4.11 | Users Profile | 83 |
| 4.12 | Get Device Registration Settings | 84 |
| 4.13 | Generate QR Code | 86 |
| 4.14 | Verify QR Code..... | 87 |
| 4.15 | Register Device for Push Notification..... | 88 |
| 4.16 | Delete Device for Push Notification..... | 89 |
| 5 | Signature Activation Data (SAD) – Body Structure | 90 |
| 6 | Get Profile Information | 91 |
| 7 | Error Code List..... | 93 |
| 8 | Use Cases of RAS Service | 99 |
| 8.1 | Restrict Access to RAS API's | 99 |

1 Introduction

1.1 Scope

This document provides information on how to integrate mobile applications and business applications with ADSS Server RAS Service for remote signature authorisation.

The integration uses REST architectural style APIs only. These calls are sent over HTTPS from the mobile device to the ADSS Server RAS Service.

1.2 Intended Readership

This guide is intended for developers who are integrating mobile applications with ADSS Server for remote signature authorisation. The document assumes a reasonable knowledge of web application development, specifically RESTful Web services and ADSS Server.

1.3 Conventions

The following typographical conventions are used in this guide to help locate and identify information:

- **Bold** text identifies menu names, menu options, items you can click on the screen, file names, folder names, and keyboard keys.
- Courier New font identifies code and text that appears on the command line.
- **Bold Courier New** identifies commands that you are required to type in.
- Courier New font identifies Ajax request/response in HTTP message body.

1.4 Technical support

If technical support is required, Ascertia has a dedicated support team. Ascertia Support can be contacted in the following ways:

Support Website www.ascertia.com/support

Support Email support@ascertia.com

Knowledge base <http://kb.ascertia.com/display/AKBS/Ascertia+Knowledge+base>

In addition to the free support service describe above, Ascertia provides formal support agreements with all product sales. Please contact sales@ascertia.com for more details.

A Product Support Questionnaire should be completed to provide Ascertia Support with further information about your system environment. When requesting help, it is always important to confirm:

- System Platform details.
- ADSS Server version number and build date.
- Details of specific issue and the relevant steps taken to reproduce it.
- Database version and patch level.
- Product log files

2 ADSS Server RAS Service Overview

ADSS Server RAS Service is the client-facing component of the ADSS Server remote signing solution. It acts as a gateway controlling access to the ADSS Server Signature Activation Module (SAM) which performs the actual remote signing operation. For brevity the ADSS Server RAS Service will be referred to as ADSS RAS throughout this document.

The purpose of ADSS RAS is to manage:

- RAS registration services:
 - Register users for remote signing. This involves not only registering the user details (e.g. name, email and phone number) but also requesting their signing key pair generation inside the ADSS Server SAM's HSM and then ensuring the corresponding public key certificate is issued by communicating with various ADSS Server components (and optionally any external CAs).
 - Register user's mobile devices for remote signing. It is possible for a user to register multiple devices.
- RAS signing services:
 - Receiving signing requests from business applications on behalf of users. Note that the business applications can either communicate with the ADSS Signing Service component which acts as a Signature Creation Application (SCA) which then passes the Data To Be Signed/Represented (DTBS/R) to ADSS RAS or they can directly interact with RAS Service.
 - Request authorisation of the remote signature from the user, by conducting a Signature Activation Protocol (SAP) with the user's registered mobile device.

Note for both registration and signing ADSS RAS is not the end-point, it acts as a front-end management service for the ADSS Server SAM service.

ADSS RAS has an Ascertia-defined API for user registration, device registration and certificate management and follows the industry-defined Cloud Signature Consortium¹ protocol for signing operations. The Signature Activation Protocol (SAP) interface with the user's mobile device for authorising the remote signature is also Ascertia-defined.

¹ See <http://www.cloudsignatureconsortium.org/> for more details

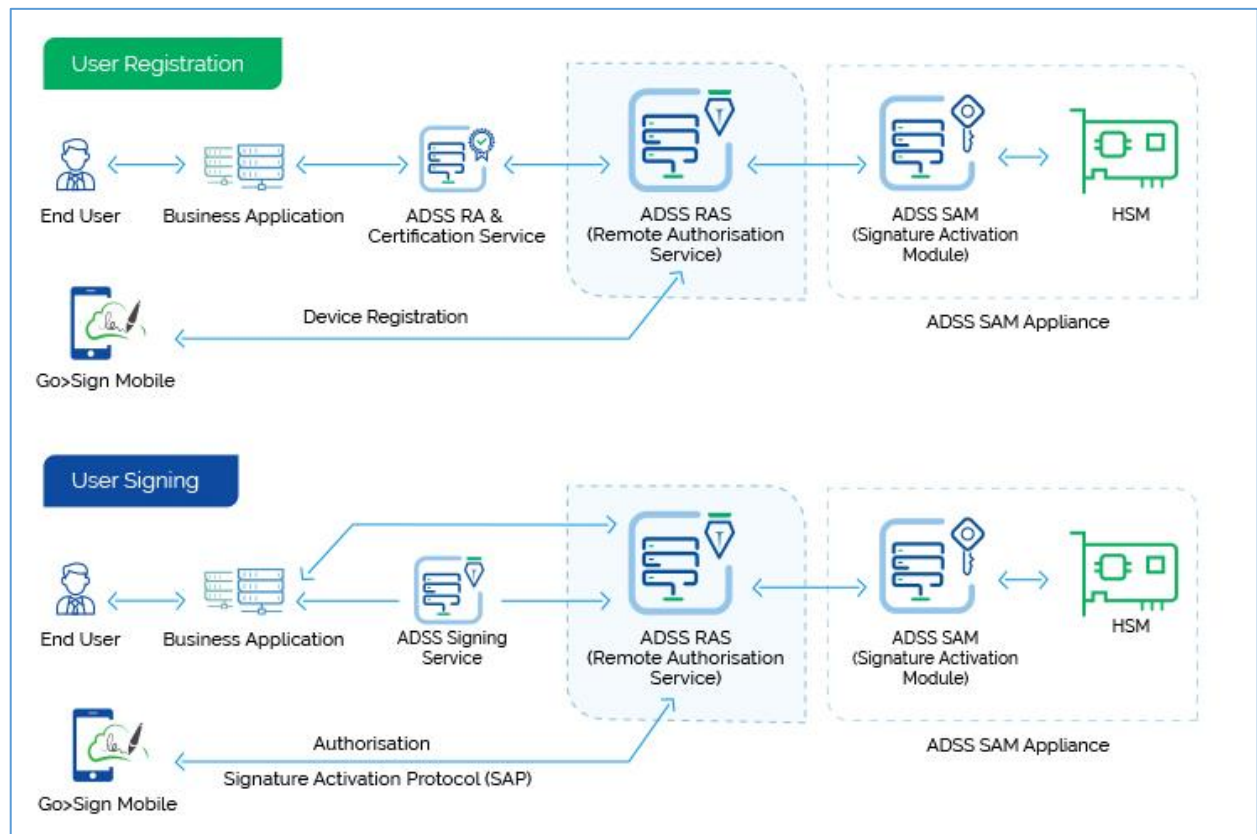


Figure 1 - RAS Service & Business Application Interaction

ADSS RAS receives all the benefits of the well-proven, robust architecture of ADSS Server. The ADSS Server Architecture & Deployment Guide describes how to implement a high availability and fault tolerant solution.

Calls to ADSS Services, including the RAS Service, use standard ADSS Server Tomcat HTTPS Listeners/Connectors. Port 8778 is used to communicate with ADSS Server over server-side TLS v1.2 and TLS v1.3.

3 Business Application Interfaces

ADSS RAS has a number of APIs aimed at business applications which initiate user registrations and signing operations. We can categorise the APIs in two sections:

- Ascertia APIs
- CSC APIs

The details of both APIs are given below:

3.1 Ascertia APIs

The APIs implemented by Ascertia for ADSS RAS Service is given below:

3.1.1 Service Status

This API is used to get the status of RAS Service whether its running, stopped or disabled. Business applications can use this API to test the connectivity with RAS Service.

| | | |
|---|-----------------------|---|
| <a href="https://<server>:8778/adss/service/ras">https://<server>:8778/adss/service/ras | | |
| HTTP Method | GET | |
| Accept | application/json | |
| Request Body | | |
| Status Code | Message | Response Body |
| 200 | OK | { "message": "success", "message_description": "ADSS RAS Service is running" } |
| 400 | Bad Request | |
| 401 | Unauthorised | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 1 – Service Status

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|--|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | A string with the description of the error_code. |

3.1.2 Register User

Creates a user in SAM Service. When a new user is created then response status '201' is returned. A business application will register its users using this interface.

| | |
|---|------|
| <a href="https://<server>:8778/adss/service/ras/v1/users">https://<server>:8778/adss/service/ras/v1/users | |
| HTTP Verb | POST |

| | | |
|--------------------|--|----------------------|
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "app_name": "Application_01", "user_name": "John Doe", "user_password": "password", "user_email": "john.doe@ascertia.com", "user_mobile": "00448007720442", "profile_id": "profile-001" }</pre> | |
| Status Code | Message | Response Body |
| 201 | Created | |
| 200 | OK | |
| 400 | Bad Request | |
| 401 | Unauthorised | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 2 – Register User

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|---|
| client_id | MANDATORY | String | Client ID which is configured in ADSS RAS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters and allowed characters are a-zA-Z0-9_@-). |
| app_name | OPTIONAL | String | Name of the organization that requested the client's Business Application to register the user. The Business Application can be dealing with multiple organizations so it can store the actual organization's information with its users to identify users of a particular organization (max. 50 characters). The same parameter can be used later as search filter in <i>Get Users API</i> . |
| user_name | MANDATORY | String | User name as friendly name for the registered user in RAS service (max. 200 characters). Following languages are supported for username: <ul style="list-style-type: none"> English Characters Norwegian Characters Slovenian Characters |

| | | | |
|---------------|-------------|--------|---|
| | | | <ul style="list-style-type: none"> • Czech & Slovak Characters • Icelandic Characters • Arabic Characters • Latvian Characters |
| user_password | CONDITIONAL | String | Password for the registered user in RAS service. Mandatory in case where Basic Authentication check is enabled in RAS profile. |
| user_email | MANDATORY | String | Email for the registered user in RAS service. It will be used to send OTP for mobile device registration etc. (max. 100 characters). |
| user_mobile | MANDATORY | String | <p>It shows the mobile number for the registered user in RAS service. The user mobile number should be provided in the following formats:</p> <ul style="list-style-type: none"> • +911234567890 • 00911234567890 <p>As mentioned above, the number must start with a '+' or '00' and the remaining part should be all digits.</p> <p>Note:</p> <ul style="list-style-type: none"> • Minimum length of the number should be at least 5 digits and maximum number can be up to 100 digits. • No characters or spaces are allowed in the mobile number. |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 100 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|--|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | A string with the description of the error_code. |

3.1.3 Update User

Updates a user's information. When a user is updated then response status '200' is returned. A business application will update its user's information using this interface.

| | |
|---|---|
| <a href="https://<server>:8778/adss/service/ras/v1/users/{client_id}/{user_id}">https://<server>:8778/adss/service/ras/v1/users/{client_id}/{user_id} | |
| HTTP Verb | PUT |
| Content-Type | application/json |
| Accept | application/json |
| Request Body | <pre>{ "user_name": "John Doe",</pre> |

| | <pre> "user_email": "john.doe@ascertia.com", "user_mobile": "00448007720442", "profile_id": "profile-001", "status": "INACTIVE" } </pre> | |
|-------------|--|---------------|
| | | |
| Status Code | Message | Response Body |
| 201 | Created | |
| 200 | OK | |
| 400 | Bad Request | |
| 401 | Unauthorised | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 3 – Update User

Request Parameters

| Parameters | Presence | Value | Description |
|-------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS RAS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters). |
| user_name | OPTIONAL | String | User name as friendly name for the registered user in RAS service (max. 50 characters). |
| user_email | MANDATORY | String | Email for the registered user in RAS service (max. 100 characters). |
| user_mobile | MANDATORY | String | <p>It shows the mobile number for the registered user in RAS service. The user mobile number should be provided in the following formats:</p> <ul style="list-style-type: none"> +911234567890 00911234567890 <p>As mentioned above, the number must start with a '+' or '00' and the remaining part should be all digits.</p> <p>Note:</p> <ul style="list-style-type: none"> Minimum length of the number should be at least 5 digits and maximum number can be up to 100 digits. No characters or spaces are allowed in the mobile number. |
| status | OPTIONAL | String | Status of the user in RAS Service. The status of a user can be updated using the values (ACTIVE/INACTIVE/BLOCKED). |

| | | | |
|------------|----------|--------|---|
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
|------------|----------|--------|---|

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|--|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | A string with the description of the error_code. |

3.1.4 Delete User

Deletes a user in RAS Service identified by {user_id}. This interface will be used by a business application to remove a user.

Exposed for: Business Applications

| https://server:8778/adss/service/ras/v1/users/{client_id}/{user_id}?profile_id=xyz | | |
|---|-----------------------|---------------|
| HTTP Verb | DELETE | |
| Accept | application/json | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 4 - Delete User

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS RAS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|------------|-------------|--------|-----------------|
| error_code | CONDITIONAL | String | The error code. |

| | | | |
|-------------------|-------------|--------|----------------------------|
| error_description | CONDITIONAL | String | Error description message. |
|-------------------|-------------|--------|----------------------------|

3.1.5 Get User

Returns a user's information registered in RAS Service identified by {user_id}. A business application will use this interface to get a user's information.

Exposed for: Business Applications

| https://server:8778/adss/service/ras/v1/users/{client_id}/{user_id}?profile_id=xyz | | |
|---|-----------------------|--|
| HTTP Verb | GET | |
| Content-Type | | |
| Accept | application/json | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "user_id": "johnDoe12" "user_name": "John Doe", "app_name": "Application_01", "user_email": "john.doe@ascertia.com", "user_mobile": "00448007720442", "status": "ACTIVE", "created_at": "2020-12-15 12:19:39", "last_updated_at": "2020-12-15 12:22:19" "profile_id": "adss:sam:profile:001", }</pre> |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 5 - Get User

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS RAS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------------|---|
| user_id | MANDATORY | <i>String</i> | User ID identifying the registered user in RAS service (max. 50 characters). |
| user_name | MANDATORY | <i>String</i> | User name as friendly name for the registered user in RAS service (max. 50 characters). |
| user_email | MANDATORY | <i>String</i> | Email for the registered user in RAS service (max. 100 characters). |
| user_mobile | MANDATORY | <i>String</i> | Mobile number for the registered user in RAS service (max. 100 characters). |
| status | OPTIONAL | <i>String</i> | Status of the user in RAS Service. The status of a user can be (ACTIVE/INACTIVE/BLOCKED). |
| created_at | MANDATORY | <i>String</i> | The date on which user is created. |
| profile_id | MANDATORY | <i>String</i> | It's a SAM Service Profile ID that is associated with the user (max. 50 characters). |
| app_name | OPTIONAL | <i>String</i> | Application name to be used by business application for listing of users. |
| last_updated_at | MANDATORY | <i>String</i> | The date on which user information is modified. |
| error | CONDITIONAL | <i>String</i> | The error code. |
| error_description | CONDITIONAL | <i>String</i> | Error description message. |

3.1.6 Get Users

Returns all the users information registered in RAS Service for a particular client identified by *{client_id}*. Paged results can also be fetched using the *start_pointer* and *fetch_size* parameters.

Exposed for: Business Applications

| | | |
|---|------------------|----------------------|
| https://server:8778/adss/service/ras/v1/users/{client_id}/{start_pointer}/{fetch_size}?{query_params} | | |
| HTTP Verb | GET | |
| Content-Type | | |
| Accept | application/json | |
| Request Body | | |
| Response Headers | | |
| x-total-records | 2 | |
| Status Code | Message | Response Body |

| | | |
|-----|-----------------------|---|
| 200 | OK | <pre> {{ "user_name": "John Doe", "user_id": "johnDoe12", "app_name": "Application_01", "user_email": "john.doe@ascertia.com", "user_mobile": "00448007720442", "status": "ACTIVE", "created_at": "2017-10-10 10:30:00", "last_updated_at": "2017-10-10 10:30:00" "profile_id": "adss:sam:profile:001" },{ "user_name": "Peter Doe", "user_id": "peterDoe12", "app_name": "Application_01", "user_email": "peter.doe@ascertia.com", "user_mobile": "00448007720442", "status": "ACTIVE", "created_at": "2017-10-10 10:30:00", "last_updated_at": "2017-10-10 10:30:00" "profile_id": "adss:sam:profile:001" }} </pre> |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 6 - Get Users

Request Parameters

| Parameters | Presence | Value | Description |
|----------------|-----------|--------|---|
| {client_id} | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| {query_params} | MANDATORY | String | <p>Currently supported parameters are:</p> <ul style="list-style-type: none"> app_name profile_id user_id user_name user_mobile user_email status <p>Response will contain all the users that fulfil the search criteria e.g. <code>.../service/ras/v1/users/my_client/0/7?profile_id=xyz&app_name=application01&user_email=user@ascertia.com&user_mobile=00448007920532&profile_id=adss:sam:profile:001&user_id=jhon123&user_name=Jhon&status=active</code></p> |

| | | | |
|---------------|-----------|---------|--|
| start_pointer | MANDATORY | Integer | Its the starting index of the data to be extracted. |
| fetch_size | MANDATORY | Integer | Its batch size client wants to fetch from RAS Service. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|---|
| user_id | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters). |
| user_name | OPTIONAL | String | User name as friendly name for the registered user in RAS service (max. 50 characters). |
| user_email | MANDATORY | String | Email for the registered user in RAS service (max. 100 characters). |
| user_mobile | MANDATORY | String | Mobile number for the registered user in RAS service (max. 100 characters). |
| status | MANDATORY | String | Status of the user in RAS Service. The status of a user can be: <ul style="list-style-type: none"> ACTIVE INACTIVE BLOCKED |
| created_at | MANDATORY | String | The date on which user is created. |
| last_updated_at | MANDATORY | String | The date on which user information is modified. |
| error | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.7 Change Password

This interface is used to change the password of a user. The user provides the old password and new password in request. The RAS verifies the old password and after successful verification, it will change the old password with the new one.

Note: This interface will only be used if a password was provided at the time of user registration, otherwise, it is of no use and the server will return error 'Unauthorized' as there will be no password stored against the user.

Exposed for: Business Applications

| | |
|---|--|
| https://server:8778/adss/service/ras/v1/users/change/password | |
| HTTP Verb | PUT |
| Content-Type | application/json |
| Accept | application/json |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "profile_id": "profile-001", "user_password": "old-password", }</pre> |

| | <code>"user_new_password": "new-password"</code> } | |
|-------------|---|--|
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | |
| 401 | Unauthorized | |
| 404 | Not Found | If URL does not contain the {Client_id} or {user_id} |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 7 - Change Password

Request Parameters

| Parameters | Presence | Value | Description |
|-------------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS RAS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
| user_password | MANDATORY | String | Old password of the user that he/she wants to change. |
| user_new_password | MANDATORY | String | New password of the user. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.8 Recover Password

Initiates user password recovery process. If a user forgets his/her password, this interface can be used to recover/reset a password. Password recovery is done in two steps; first the business application will call this interface to initiate the process, then RAS will send either one or two OTPs to user's mobile and email according to the RAS Profile settings. The client will send these OTPs in a separate call using another interface discussed in next section.

Note: If the user was registered without password, this interface can also be used to set a password for that user.

Exposed for: Business Applications

<https://server:8778/adss/service/ras/v1/users/password/recover>

| | | |
|--------------------|--|---|
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "profile_id": "profile-001" }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | If two OTPs will be sent to user: <pre>[{ "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }, { "type": "SMS_OTP", "sent_to": "+448007720442" }]</pre> |
| | | If one OTP will be sent on user email: <pre>{ "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }</pre> |
| | | If one OTP will be sent to user's mobile: <pre>{ "type": "SMS_OTP", "sent_to": "+448007720442" }</pre> |
| 400 | Bad Request | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 8 – Recover Password

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS RAS Console > Client Manager (max. 50 characters). |

| | | | |
|------------|-----------|--------|---|
| user_id | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|---|
| type | CONDITIONAL | String | As OTP can be sent on both mediums i.e email/mobile so it defines the type of OTP. There can be following types: <ul style="list-style-type: none"> - EMAIL_OTP - SMS_OTP |
| sent_to | CONDITIONAL | String | It could be the mobile number or email of the user depends upon the "type" of OTP. |
| error | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.9 Confirm Recover Password

Completes user password recovery process. The business application will send the OTPs and new password in request using this API, and the RAS will first validate the OTP and after successful validation, change the old password with the new password.

Exposed for: Business Applications

| | | |
|---|--|----------------------|
| https://server:8778/adss/service/ras/v1/users/password/recoverconfirm | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "profile_id": "profile-001", "sms_otp": "225665", "email_otp": "654456", "user_password": "P@\$w0rD!@" }</pre> <p>If only one OTP will be received by user either on SMS or email, the request would contain only "sms_otp" or "email_otp".</p> | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | |

| | | |
|-----|-----------------------|--|
| 401 | Unauthorized | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 9 – Confirm Recover Password

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
| sms_otp | MANDATORY | String | OTP received on the user's registered mobile number. |
| email_otp | MANDATORY | String | OTP received on the user's registered email. |
| user_password | MANDATORY | String | New password for the registered user. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.10 Change User Email

This interface will be used by a business application to change a user's email. The change email process completes in two steps. In first step, the business application will send the user ID and new email address on this interface, and the RAS will send the OTP(s) to the user (one on mobile and one on the email address according to RAS Profile). The business application will then send these OTPs in another request using another interface that is discussed in next section.

Note: A user's email can also be changed using the "Update User" API but this API provides more security and control while changing the email. The clients can choose the relevant APIs to change a user's email according to their requirements and policies.

Exposed for: Business Applications

| | |
|---|------------------|
| https://server:8778/adss/service/ras/v1/users/email/change | |
| HTTP Verb | POST |
| Content-Type | application/json |
| Accept | application/json |

| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "user_email": "john.doe@ascertia.com", "profile_id": "profile-001" }</pre> | |
|--------------|---|---|
| Status Code | Message | Response Body |
| 200 | OK | <p>If two OTPs will be sent to user:</p> <pre>[{ "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }, { "type": "SMS_OTP", "sent_to": "+448007720442" }]</pre> |
| | | <p>If one OTP will be sent on user email:</p> <pre>{ "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }</pre> |
| | | <p>If one OTP will be sent to user's mobile:</p> <pre>{ "type": "SMS_OTP", "sent_to": "+448007720442" }</pre> |
| 400 | Bad Request | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 10 - Change User Email

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |

| | | | |
|------------|-----------|--------|---|
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
| user_email | MANDATORY | String | New Email for the registered user (max. 100 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|---|
| type | MANDATORY | String | Type of the OTP e.g. sms/email. |
| sent_to | MANDATORY | String | Mobile number or email of the user where OTP is sent. |
| error | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.11 Confirm Change User Email

Once the OTPs are received by the user for change email, the business application will provide these OTPs to RAS by calling this interface. The RAS will first validate the both OTPs and then change the old email with the new email address.

Exposed for: Business Applications

| | | |
|---|---|----------------------|
| https://server:8778/adss/service/ras/v1/users/email/changeconfirm | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "sms_otp": "225665", "email_otp": "654456", "profile_id": "profile-001" }</pre> <p>If only one OTP will be received by user either on SMS or email, the request would contain only "sms_otp" or "email_otp".</p> | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | |
| 401 | Unauthorized | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 11 – Confirm Change User Email

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
| sms_otp | MANDATORY | String | OTP received by the user's registered mobile number. |
| email_otp | MANDATORY | String | OTP received by the user's registered email. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|--|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | A string with the description of the error_code. |

3.1.12 Change User Mobile

A business application will call this interface of RAS in order to change a user's mobile number. Like 'Change Email', this process also completes in two steps. The business application will send the user ID and new mobile number on this interface and RAS will send the OTPs (one on user's email and another on the provided new mobile number).

After receiving the OTPs, the business application will call another interface discussed in next section to complete the process.

Note: The user's mobile number can also be changed using the "Update User" API, but this API provides more security and control and clients can choose the preferred API according to the requirements.

Exposed for: Business Applications

| | |
|---|---|
| https://server:8778/adss/service/ras/v1/users/mobile/change | |
| HTTP Verb | POST |
| Content-Type | application/json |
| Accept | application/json |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "user_mobile": "00448007720442", "profile_id": "profile-001" }</pre> |

| Status Code | Message | Response Body |
|-------------|-----------------------|---|
| 200 | OK | If two OTPs will be sent to user: [{ "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }, { "type": "SMS_OTP", "sent_to": "+448007720442" }] |
| | | If one OTP will be sent on user email: { "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", } |
| | | If one OTP will be sent to user's mobile: { "type": "SMS_OTP", "sent_to": "+448007720442" } |
| 400 | Bad Request | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 12 - Change User Mobile

Request Parameters

| Parameters | Presence | Value | Description |
|-------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
| user_mobile | MANDATORY | String | New mobile number for the registered user (max. 100 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------------|---|
| type | MANDATORY | <i>String</i> | As OTP can be sent on both mediums i.e email/mobile so it defines the type. |
| sent_to | MANDATORY | <i>String</i> | It could be the mobile number or email of the user depends upon the OTP type. |
| error | CONDITIONAL | <i>String</i> | The error code. |
| error_description | CONDITIONAL | <i>String</i> | Error description message. |

3.1.13 Confirm Change User Mobile

Once the OTPs to change user mobile are received by the user. The business application will call this interface providing the both OTPs to RAS Service. The RAS Service will first validate the OTPs and after successful verification, it will change the old mobile number with the new one.

Exposed for: Business Applications

| https://server:8778/adss/service/ras/v1/users/mobile/changeconfirm | | |
|---|---|----------------------|
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "sms_otp": "225665", "email_otp": "654456", "profile_id": "profile-001" }</pre> <p>If only one OTP will be received by user either on SMS or email, the request would contain only "sms_otp" or "email_otp".</p> | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | |
| 401 | Unauthorized | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 13 – Confirm Change User Mobile

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
| sms_otp | MANDATORY | String | OTP received by the user's registered mobile number. |
| email_otp | MANDATORY | String | OTP received by the user's registered email. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.14 Get Registered Devices

This API is used to get a list of all the registered mobile devices of a user.

Exposed for: Business Applications

| | | |
|---|--|---|
| https://server:8778/adss/service/ras/v1/users/devices | | |
| HTTP Verb | POST | |
| Content-Type | | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "profile_id": "profile-001", "user_id": "johnDoe12" }</pre> | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>[{ "device_id": "2eb1846d-81d8-40d0-86ba-d20bdf7ac5e0", "device_name": "iPhone", "secure_element": true, "biometric": true, }, { "device_id": "3fc29573-92e9-40d0-86ba-d20bdf7ac5e0",</pre> |

| | | |
|-----|-----------------------|--|
| | | <pre>"device_name": "Samsung", "secure_element": true, "biometric": true, }]</pre> |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 14 – Get Registered Devices

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------|---|
| device_id | MANDATORY | String | Device ID which is created at the time of device registration. |
| device_name | MANDATORY | String | Device name which is set at the time of device registration. |
| secure_element | MANDATORY | Boolean | “True” if device has secure element/enclave. |
| biometric | MANDATORY | Boolean | “True” if device has biometric feature available on the device. It can be TouchID, FaceID, Fingerprint etc. |
| user_id | MANDATORY | String | User ID identifying the user the device belongs to. |
| error | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.15 Delete Device

Deletes a user's mobile device in RAS Service identified by {user_id} and {device_id}.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/devices/{client_id}/{user_id}/{device_id}?{profile_id}=xyz

| | | |
|--------------------|-----------------------|----------------------|
| HTTP Verb | DELETE | |
| Accept | application/json | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 15 - Delete Device

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|--|
| {client_id} | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| {user_id} | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| {device_id} | MANDATORY | String | Device ID identifying the mobile device |
| {profile_id} | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.16 Generate Key Pair

Creates a key pair for the user in RAS Service. This key pair will be used to sign the documents. When a new key pair is created the response status '201' is returned.

Exposed for: Business Applications

| | |
|---|--|
| https://server:8778/adss/service/ras/v1/keypairs | |
| HTTP Verb | POST |
| Content-Type | application/json |
| Accept | application/json |
| Request Body | { "client_id": "samples_test_client", |

| | <pre> "user_id": "johnDoe12", "app_name": "Application_01", "user_password": "*****", "key_alias": "sample_key_alias", "profile_id": "adss:ras:profile:001" } </pre> | |
|-------------|--|---------------|
| Status Code | Message | Response Body |
| 201 | Created | |
| 200 | OK | |
| 400 | Bad Request | |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 16 – Generate Key Pair

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user for whom the key pair is being generated (max. 50 characters). |
| app_name | OPTIONAL | String | Name of the organization that requested the client's Business Application to generate the key-pair for a user. The Business Application can be dealing with multiple organizations so it can store the actual organization's information with the key-pair to identify keys of a particular organization. (max. 50 characters). |
| user_password | OPTIONAL | String | Password will only be required if key wrapping with Dynamic KEK is enabled in Hardware Crypto Profile. |
| key_alias | MANDATORY | String | Key Alias to identify the key pair. |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.17 Delete Key Pair

Deletes a user's key-pair in RAS Service identified by {user_id} and {key_alias}. The business applications will call this interface to delete a key-pair of a user.

Exposed for: Business Applications

| | | |
|---|-----------------------|----------------------|
| https://server:8778/adss/service/ras/v1/keypairs/{client_id}/{user_id}/{key_alias}?profile_id={profile_id} | | |
| HTTP Verb | DELETE | |
| Accept | application/json | |
| Request Body | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 17 – Delete Key Pair

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|--|
| {client_id} | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| {user_id} | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| {key_alias} | MANDATORY | String | Key Alias of the key pair that is going to be deleted. |
| {profile_id} | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.18 Get CSR

Returns the base64 encoded CSR (Certificate Signing Request i.e. PKCS#10) of the key pair generated for a user. The business applications will call this interface to get a CSR after generating a key-pair for a

user. The client will get this CSR certified and provide the certificate to RAS using the "Import Certificate" API discussed next.

Exposed for: Business Applications

| https://server:8778/adss/service/ras/v1/keypairs/csr | | |
|---|--|---|
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "user_password": "password12", "key_alias": "sample_key_alias", "profile_id": "profile-001" }</pre> | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "csr": " MIICUzCCATsCAQAwDjEMMAoGA1.....KJh" }</pre> |
| 400 | Bad Request | |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 18 - Get CSR

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|--|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| user_password | OPTIONAL | String | Used to unwrap the user key if the key was wrapped with a dynamic KEK during generation. |
| key_alias | MANDATORY | String | Key Alias of key pair for which CSR to be generated. |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-----------|---------------|----------------------------|
| csr | MANDATORY | <i>String</i> | Base 64 encoded CSR. |
| error_code | MANDATORY | <i>String</i> | The error code. |
| error_description | MANDATORY | <i>String</i> | Error description message. |

3.1.19 Import Certificate

Uploads or import the user's certificate and certificate chain related to a key of the user. This certificate will be stored against its relevant key-pair. A certificate must be imported to RAS Service cause a key cannot be used for signing if not certified, so certificate provides the proof the key has been certified.

Exposed for: Business Applications

| https://server:8778/adss/service/ras/v1/keypairs/cert | | |
|---|---|---------------|
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "samples_test_client", "user_id": "johnDoe12", "key_alias": "sample_key_alias", "profile_id": "profile-001", "certificate": "HyguhugyCATsCAQAwDjEMMAoGA1.....jhgh=", "certificate_chain": ["HyguhugyCATsCAQAwDjEMMAoGA1.....jhgh=", "HyguhugyCATsCAQAwDjEMMAoGA1.....jhgh=", ...], "p7b": "HyguhugyCATsCAQAwDjEMMAoGA1.....jhgh=" }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 19 - Import Certificate

Request Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|---|
| client_id | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | String | User ID identifying the registered user (max. 50 characters). |
| key_alias | MANDATORY | String | Key Alias of key pair for which certificate is being imported. |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |
| certificate | MANDATORY | String | Base 64 encoded certificate. |
| certificate_chain | CONDITIONAL | Array | Array containing certificates chain in Base 64 encoded string. |
| p7b | CONDITIONAL | String | Certificate chain can also be provided in p7b format as Base 64 encoded string. certificate_chain and p7b can be provided alternatively. If both are present p7b will override certificate_chain. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.20 Get User's Certificates

Returns a list of all the certificates (with chains) for the provided registered user.

Exposed for: Business Applications

| | | |
|---|------------------|--|
| https://server:8778/adss/service/ras/v1/keypairs/cert/{client_id}/{user_id}?profile_id=adss:ras:profile:01 | | |
| HTTP Verb | GET | |
| Content-Type | | |
| Accept | application/json | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | [{ "key_status": "ACTIVE", "app_name": "Application_01" "key_alias": "sample_cert_alias_01", "user_id": "Alice", "certificate_chain": [|

| | | |
|-----|-----------------------|---|
| | | <pre> {"HyguhugyCATsCAQAwDjEMMAoGA1.....jhgjh="}, {"HyguhugyCATsCAQAwDjEMMAoGA1.....jhgjh="} ...] }, { "user_id": "johnDoe12", "key_alias": "sample_cert_alias_02", "app_name": "Application_01" "key_status": " ACTIVE ", "certificate_chain": [{"HyguhugyCATsCAQAwDjEMMAoGA1.....jhgjh="}, {"HyguhugyCATsCAQAwDjEMMAoGA1.....jhgjh="} ...] }} </pre> |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 20 - Get User Certificates

Request Parameters

| Parameters | Presence | Value | Description |
|----------------|-----------|--------|--|
| {client_id} | MANDATORY | String | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| {user_id} | MANDATORY | String | User ID identifying the registered user in RAS service (max. 50 characters). |
| {query_params} | MANDATORY | String | <p>Currently supported parameters are:</p> <ul style="list-style-type: none"> profile_id app_name <p>Response will contain the certificates that contain this app_name provided as query parameter e.g. .../service/ras/v1/keypairs/cert/list/my_client_01/user_01?client_id=abc&profile_id=xyz&app_name=application01</p> |

Response Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|---|
| user_id | MANDATORY | String | User ID identifying the registered user |
| app_name | OPTIONAL | String | Application name that was provided by business application while generating the key pair. |
| key_alias | MANDATORY | String | Key Alias of key pair the certificate belongs to. |

| | | | |
|-------------------|-------------|--------|--|
| key_status | MANDATORY | String | Status of the key i.e. Active or Inactive |
| certificate | MANDATORY | String | Base 64 encoded string representing the certificate. |
| certificate_chain | MANDATORY | Array | Array containing certificates chain in Base 64 encoded string. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.1.21 Authentication/Login without Password

User can be registered without password so this API can be used to authenticate a user using client credentials.

| | | |
|---|--|---|
| <a href="https://<server>:8778/adss/service/ras/v1/login">https://<server>:8778/adss/service/ras/v1/login | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre>{ "client_id": "adss...client", "client_secret": "fj49kl.....oOpQS", "profile_id": "ADSS RAS Profile 001", "user_id": jhon.wick }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXZWQ", "expires_in": 3600 }</pre> |
| 400 | Bad Request | <pre>{ "error": "58071", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |
| 401 | Unauthorised | <pre>{ "error": "59033", "error_description": "Failed to process request - user ID or password is invalid" }</pre> |

Table 21 – Authentication/Login

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|---------------|--|
| client_id | MANDATORY | <i>String</i> | Client ID which is configured in ADSS Console > Client Manager (max. 50 characters). |
| user_id | MANDATORY | <i>String</i> | User ID identifying the registered user (max. 50 characters). |
| client_secret | MANDATORY | <i>String</i> | Secret of the client used to authenticate it. |
| profile_id | OPTIONAL | <i>String</i> | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------------|--|
| access_token | MANDATORY | <i>String</i> | It will return the client access token after the authentication of client ID and secret. |
| expires_in | MANDATORY | <i>String</i> | Token expiry in seconds. |
| error_code | CONDITIONAL | <i>String</i> | The error code. |
| error_description | CONDITIONAL | <i>String</i> | Error description message. |

3.2 CSC APIs

Ascertia has implemented CSC protocol to perform remote authorised signing. The Cloud Signature Consortium (CSC) is a group of industry and academic organizations committed to building new standards for cloud-based digital signatures that will support web and mobile applications and comply with the most demanding electronic signature regulations in the world.

Ascertia RAS Service supports CSC v1 APIs according to the specification version (1.0.4.0). For complete details of the APIs and parameters, please refer to the CSC specification for the same version (1.0.4.0).

3.2.1 Authentication/Login

It is a username and password based authentication call which after successful authentication returns an access token and optionally refresh token based on input parameter in request.

| | | |
|---|--|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/auth/login">https://<server>:8778/adss/service/ras/csc/v1/auth/login | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Basic c2FuOnBhc3N3b3Jk... This is the base64 encoded value of Username:UserPassword. | |
| Request Body | <pre>{ "client_id": "adss...client", "client_secret": "fj49kl.....oOpQS", "profile_id": "ADSS RAS Profile 001", "rememberMe": true }</pre> | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpzZW50L3VzZXQ", "refresh_token": "eyJpc3MinRpYSN1Yil6InN1YiI6IjE5OTkxMjM0NTY3ODk", "expires_in": 3600 }</pre> |
| 400 | Bad Request | <pre>{ "error": "58039", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |
| 401 | Unauthorised | <pre>{ "error": "59033", "error_description": "Failed to process request - user ID or password is invalid" }</pre> |

Table 1 – Authentication/Login

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-------------|---------|--|
| client_id | MANDATORY | String | The unique ' <i>client_id</i> ' previously assigned to the signature application by remote service. |
| client_secret | MANDATORY | String | The ' <i>client_secret</i> ' shall be passed if no authorization header and no client assertion is used. |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request. This is Ascertia's custom parameter. |
| refresh_token | CONDITIONAL | String | The long-lived refresh token returned from a previous call to this method with HTTP Basic Authentication. This may be used as an alternative to the Authorization header to re-authenticate the user according to the method described in RFC 6749. In such case, the encoded userId and password shall not be provided in the HTTP Authorization header. Note: This refresh token may not be compatible with refresh tokens obtained by means of OAuth 2.0 authorization. |
| remember_me | OPTIONAL | Boolean | A Boolean value typically corresponding to an option that the user may activate during the authentication phase to 'stay signed in' and maintain a valid authentication across multiple sessions: <ul style="list-style-type: none"> • True: If the remote service supports user reauthentication, a <i>refresh_token</i> will be returned and the signature application may use it on a subsequent call to this method instead of passing an Authorization header. • False: A '<i>refresh_token</i>' will not be returned. If the parameter is omitted, it will default to 'false'. |
| clientData | OPTIONAL | String | Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes. Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully. |

Response Parameters

| Parameters | Presence | Value | Description |
|---------------|-------------|--------|--|
| access_token | MANDATORY | String | The short-lived services access token used to authenticate the subsequent API requests within the same session. This token shall be used as the value of the 'Authorization: Bearer' in the HTTP header of the API requests. When receiving an API call with an expired token, the remote services shall return an error and require a new auth/login request. |
| refresh_token | CONDITIONAL | String | The long-lived refresh token used to re-authenticate the user on the subsequent session. The value is returned if the |

| | | | |
|-------------------|-------------|--------|--|
| | | | <i>rememberMe</i> parameter in the requests is 'true' and the remote service supports user authentication. |
| expires_in | OPTIONAL | String | The lifetime in seconds of the service access token. If omitted, the default expiration time is 3600 seconds (1 hour). |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.2.2 Authentication/Revoke

Revoke a service access token or refresh token that was obtained from the RAS Service. This method exists to enforce the security of the RAS Service. When the Business Application needs to terminate a session, it is recommended to invoke this method to prevent further access by reusing the token.

| | | |
|---|--|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/auth/revoke">https://<server>:8778/adss/service/ras/csc/v1/auth/revoke | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| Request Body | <pre>{ "token": "_TiHRG-bA H3XIFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw", "token_type_hint": "refresh_token" }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |
| | | <pre>{ "error": "invalid_request", "error_description": "Invalid string parameter token_type_hint" }</pre> |
| | | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter token" }</pre> |
| | | <pre>{</pre> |

| | | |
|--|--|--|
| | | <pre>"error": "invalid_request", "error_description": "Invalid string parameter token" }</pre> |
|--|--|--|

Table 2 – Authentication/Revoke

Request Parameters

| Parameters | Presence | Value | Description |
|-----------------|-----------|---------------|---|
| token | MANDATORY | <i>String</i> | The token that the signature application wants to get revoked. |
| token_type_hint | OPTIONAL | <i>String</i> | An optional hint about the type of the token submitted for revocation. If the parameter is omitted, the RAS Service will identify the token across all the available tokens. |
| clientData | OPTIONAL | <i>String</i> | Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes. Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------------|----------------------------|
| error_code | CONDITIONAL | <i>String</i> | The error code. |
| error_description | CONDITIONAL | <i>String</i> | Error description message. |

3.2.3 Credentials/List

Returns the list of credentials associated with a user identifier. A user may have one or multiple credentials.

| | | |
|---|------------------------------------|----------------------|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/credentials/list">https://<server>:8778/adss/service/ras/csc/v1/credentials/list | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| Request Body | <pre>{ "userID": "Jhon", }</pre> | |
| | | |
| Status Code | Message | Response Body |

| | | |
|-----|-------------|---|
| 200 | OK | <pre>{ "credentialIDs": ["johnDoe"] }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |

Table 3 – Credentials/List

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-------------|--------|---|
| userID | CONDITIONAL | String | The identifier associated to the identity of the credential owner. This parameter shall not be present if the service authorization is user-specific. In that case, the userID is already implicit in the service access token passed in the Authorization header. If service access token is obtained, Client Credential's Flow then userID is not part of the access token. In such case this parameter should be provided. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|---|
| credentialIDs | MANDATORY | String | One or more credentialID(s) associated with the provided or implicit userID. No more than <i>maxResults</i> items shall be returned. |
| nextPageToken | OPTIONAL | String | The page token required to retrieve the next page of results. No value shall be returned if the remote service does not support items pagination or the response relates to the last page of results. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.2.4 Credentials/Info

Retrieve the credential and return the main identity information and the public key certificate or the certificate chain associated to it.

<https://<server>:8778/adss/service/ras/csc/v1/credentials/info>

| | | |
|--------------------|---|--|
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| Request Body | <pre>{ "credentialID": "JohnDoe", "certificates": "chain", "certInfo": true, "authInfo": true }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "description": "Go>Sign mobile based implicit credential authorization", "key": { "status": "enabled", "algo": ["1.2.840.113549.1.1.1"], "len": 2048, "curve": "1.2.840.10045.3.1.7", }, "cert ": { "status": "valid", "certificates": ["MIIFDzCCNI0IRV+Vbe132oyYm1dzz9GI1VqEiPqaKc8miP Wb1ssF+MNylyBk\r\n7qnt", "MIIFDzCCNI0IRV+Vbe132oyYm1dzz9GI1VqEiPqaKc8miP Wb1ssF+MNylyBk\r\n7qnt", "MIIEhZCCy2VCpgcCp1xOLhqp+A3TF/8c07EEDjhcmK OVEFz\r\nscEsJHThGj2/buU="], "issuerDN": "CN=ADSS Samples Test CA,OU=Ascertia Software Distribution,O=Ascertia Limited,C=GB", "SerialNumber": "214548948485166938134883584853795496857900189230", "subjectDN": "CN=za,OU=Development,O=Ascertia,C=GB", "validFrom": "20201215112008+0000", "validTo": "20211215112008+0000", } }</pre> |

| | | |
|-----|-------------|--|
| | | <pre> }, "authMode": "explicit", "PIN": { "presence": "true", "format": "A", "label": "PIN", "description": "Please enter the signature PIN" }, "OTP": { "presence": "true", "type": "online", "format": "N", "label": "Mobile OTP", "description": "Please enter the 6 digit code you received by SMS" }, "SCAL": "2", "multisign": 1, "lang": "en-GB" } </pre> |
| 400 | Bad Request | <pre> { "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." } </pre> |
| 400 | Bad Request | <pre> { "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter credentialID" } </pre> |
| 400 | Bad Request | <pre> { "error": "58100", "error_description": "Invalid parameter credentialID" } </pre> |

Table 4 – Credentials/Info

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|--|
| credentialID | MANDATORY | String | The unique identifier associated to the credential. |
| certificates | OPTIONAL | String | Specific which certificates from the certificate chain shall be returned in cert/certificates. |

| | | | |
|------------|----------|----------------|--|
| | | | <ul style="list-style-type: none"> • None: No certificate shall be returned. • Single: Only the end entity certificate shall be returned. • Chain: The full certificate chain shall be returned. <p>The default value is 'single', so if the parameter is omitted then the method will only return the end entity certificate.</p> |
| certInfo | OPTIONAL | <i>Boolean</i> | <p>Request to return various parameters containing information from the end entity certificate. This is useful in case the business application wants to retrieve some details of the certificate without having to decode it first.</p> <p>The default value is 'false', so if the parameter is omitted then the information will not be returned.</p> |
| authInfo | OPTIONAL | <i>Boolean</i> | <p>Request to return various parameters containing information on the authorization mechanisms supported by this credential.</p> <p>The default value is 'false', so if the parameter is omitted then the information will not be returned.</p> |
| lang | OPTIONAL | <i>String</i> | <p>Request a preferred language of the response to the remote service, specified according to RFC 5646.</p> <p>If present, the remote service shall provide language-specific responses using the specified language. If the specified language is not supported then it shall provide these responses in the language as specified in the lang output parameter.</p> |
| clientData | OPTIONAL | <i>String</i> | <p>Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes.</p> <p>Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully.</p> |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------|-----------|---------------|---|
| description | OPTIONAL | <i>String</i> | A free form description of the credential in the <i>lang</i> language. The maximum size of the string is 255 characters. |
| key/status | MANDATORY | <i>String</i> | <p>The status of the signing key of the credential are as follows:</p> <ul style="list-style-type: none"> • Enabled: If the signing key is enabled, it can be used for signing. |

| | | | |
|-------------------|-------------|--------|---|
| | | | <ul style="list-style-type: none"> Disabled: If the signing key is disabled, it cannot be used for signing. This may occur when the operator has disabled it or when it has been detected that the associated certificate is expired or revoked. |
| key/algo | MANDATORY | String | <p>The list of OIDs of the supported key algorithms. For example:</p> <ul style="list-style-type: none"> 1.2.840.113549.1.1.1=RSA encryption 1.2.840.10045.4.3.2=ECDSA with SHA256 |
| key/len | MANDATORY | Number | The length of the cryptographic key in bits. |
| key/curve | CONDITIONAL | String | The OID of the ECDSA curve. The value shall only be returned if <i>keyAlgo</i> is based on ECDSA. |
| cert/status | OPTIONAL | String | The status of validity of the end entity certificate. The value is OPTIONAL, so the RAS Service will only return a value that is accurate and consistent with the actual validity status of the certificate at the time the response is generated. |
| cert/certificates | CONDITIONAL | String | One or more Base64-encoded X.509v3 certificates from the certificate chain. If the certificates parameter is “ chain ”, the entire certificate chain shall be returned with the end entity certificate at the beginning of the array. If the certificates parameter is “ single ”, only the end entity certificate shall be returned. If the certificates parameter is “ none ”, this value shall not be returned. |
| cert/issuerDN | CONDITIONAL | String | The Issuer Distinguished Name from the X.509v3 end entity certificate as UTF-8-encoded character string according to RFC 4514. This value shall be returned when <i>certInfo</i> is “ true ”. |
| cert/serialNumber | CONDITIONAL | String | The Serial Number from the X.509v3 end entity certificate represented as hex-encoded string format. This value shall be returned when <i>certInfo</i> is “ true ”. |
| cert/subjectDN | CONDITIONAL | String | The Subject Distinguished Name from the X.509v3 end entity certificate as UTF-8-encoded character string, according to RFC 4514. This value shall be returned when <i>certInfo</i> is “ true ”. |
| cert/validFrom | CONDITIONAL | String | The validity start date from the X.509v3 end entity certificate as character string, encoded as GeneralizedTime (RFC 5280) (e.g. “YYYYMMDDHHMMSSZ”). This value shall be returned when <i>certInfo</i> is “ true ”. |
| cert/validTo | CONDITIONAL | String | The validity end date from the X.509v3 end entity certificate as character string, encoded as GeneralizedTime (RFC 5280) (e.g. |

| | | | |
|--------------|-------------|--------|---|
| | | | “YYYYMMDDHHMMSSZ”). This value shall be returned when <i>certInfo</i> is “true”. |
| authMode | MANDATORY | String | <p>Specifies one of the authorization modes from below:</p> <ul style="list-style-type: none"> • Implicit: The authorization process is managed by the RAS Service autonomously. Authentication factors are managed by the RAS by interacting directly with the user, and not by the business application. • OAuth2code: The authorization process is managed by the RAS Service using an OAuth 2.0 mechanism. • Explicit: The authorization process is managed by the signature application, which collects authentication factors like PIN or One-Time Passwords (OTP). |
| SCAL | OPTIONAL | String | <p>Specifies if the RAS Service will generate for this credential a signature activation data (SAD) that contains a link to the hash to-be-signed:</p> <ul style="list-style-type: none"> • “1”: The hash to-be-signed is not linked to the signature activation data. • “2”: The hash to-be-signed is linked to the signature activation data. <p>This value is OPTIONAL and the default value is “1”.</p> <p>NOTE: The difference between SCAL1 and SCAL2, as described in CEN TS 119 241-1 [i.5], is that for SCAL2, the signature activation data needs to have a link to the data to-be-signed. The value “2” only gives information on the link between the hash and the SAD, it does not give information if a full SCAL2 as described in CEN TS 119 241-1 [i.5] is implemented.</p> <p>NOTE: RAS Service always returns “2” for this parameter.</p> |
| PIN/presence | CONDITIONAL | String | Specifies if a text-based PIN is required, forbidden, or optional. This value shall be present only when <i>authMode</i> is “explicit”. |
| PIN/format | CONDITIONAL | String | <p>The data format of the PIN string:</p> <ul style="list-style-type: none"> • ‘A’: The PIN string contains alphanumeric text and/or symbols like “-.%!\$@#+”. • ‘N’: The PIN string only contains numeric text. <p>This value SHALL be present only when <i>authMode</i> is “explicit” and PIN/presence is not “false”.</p> |

| | | | |
|-----------------|-------------|---------------|--|
| | | | NOTE: The size of the expected PIN is not specified, since this information could help an attacker in performing guessing attacks. |
| PIN/label | CONDITIONAL | <i>String</i> | A label for the data field used to collect the PIN in the user interface of the signature application, in the language specified in the lang parameter. This value may be present only when <i>authMode</i> is “explicit” and PIN/presence is not “false”. The maximum size of the string is 255 characters. |
| PIN/description | CONDITIONAL | <i>String</i> | A free form description of the PIN in the language specified in the lang parameter. This value may be present only when <i>authMode</i> is “explicit” and PIN/presence is not “false”. The maximum size of the string is 255 characters. |
| OTP/presence | CONDITIONAL | <i>String</i> | Specifies if a text-based One-Time Password (OTP) is required, forbidden, or optional. This value shall be present only when <i>authMode</i> is “explicit”. |
| OTP/type | CONDITIONAL | <i>String</i> | <p>The type of the OTP:</p> <ul style="list-style-type: none"> • Offline: The OTP is generated offline by a dedicated device and does not require the client to invoke the credentials/sendOTP method. • Online: The OTP is generated online by the remote service when the client invokes the credentials/sendOTP method. <p>This value shall be present only when <i>authMode</i> is “explicit” and OTP/presence is not “false”.</p> |
| OTP/format | CONDITIONAL | <i>String</i> | <p>The data format of the OTP string:</p> <ul style="list-style-type: none"> • ‘A’: The OTP string contains alphanumeric text and/or symbols like “-.%!\$@#+”. • ‘N’: The OTP string only contains numeric text. <p>This value shall be present only when <i>authMode</i> is “explicit” and OTP/presence is not “false”.</p> |
| OTP/label | CONDITIONAL | <i>String</i> | A label for the data field used to collect the OTP in the user interface of the signature application, in the language specified in the lang parameter. This value may be present only when <i>authMode</i> is “explicit” and OTP/presence is not “false”. The maximum size of the string is 255 characters. |
| OTP/description | CONDITIONAL | <i>String</i> | A free form description of the OTP mechanism in the language specified in the lang parameter. This value may be present only when <i>authMode</i> is “explicit” and OTP/presence is not “false”. |

| | | | |
|-------------------|-------------|--------|---|
| | | | The maximum size of the string is 255 characters. |
| OTP/ID | CONDITIONAL | String | The identifier of the OTP device or application. This value shall be present only when <i>authMode</i> is “explicit” and OTP/presence is not “false”. |
| OTP/provider | CONDITIONAL | String | The provider of the OTP device or application. This value MAY be present only when <i>authMode</i> is “explicit” and OTP/presence is not “false”. |
| multisign | MANDATORY | Number | A number equal or higher to 1 representing the maximum number of signatures that can be created with this credential with a single authorization request. |
| lang | OPTIONAL | String | The language used in the responses, specified according to RFC 5646. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.2.5 Credentials/Authorize

Authorize the access to the credential for remote signing, according to the authorization mechanisms associated to it. This method returns the [Signature Activation Data \(SAD\)](#) required to authorize the signatures/signHash method.

| | |
|---|--|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/credentials/authorize">https://<server>:8778/adss/service/ras/csc/v1/credentials/authorize | |
| HTTP Verb | POST |
| Content-Type | application/json |
| Accept | application/json |
| Authorization | Bearer {access_token} |
| Request Body | <pre>{ "credentialID": "JohnDoe", "numSignatures": 2, "documents": [{ "document_id": 123, "document_name": "Document Name 123", },{ "document_id": 456, "document_name": "Document Name 456", }], "hash": ["sTOgwOm+474gFj0q0x1iSNspKqbcse4leiqlDg/HWul=", "c1RPZ3dPbSs0NzRnRmowcTB4MWITTnNwS3FiY3NINEllaXFsRGcvSFd1ST0="], "PIN": "12345678", "OTP": "738496" }</pre> |

| | Note: The 'documents', 'document_id' and 'document_name' are Ascertia's custom parameters and are optional in this case. | |
|-------------|---|---|
| | | |
| Status Code | Message | Response Body |
| 200 | OK | { "SAD": "_TiHRG-bA4/CKN69L8gdSYp5_pw" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter credentialID" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid parameter credentialID" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Missing (or invalid type) integer parameter numSignatures" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid parameter numSignatures" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid PIN parameter - Failed to authenticate PIN" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid OTP parameter - Failed to authenticate OTP" } |

Table 5 – Credentials/Authorize

Request Parameters

| Parameters | Presence | Value | Description |
|-------------------------|-------------|---------------|---|
| credentialID | MANDATORY | <i>String</i> | The unique identifier associated to the credential. |
| numSignatures | MANDATORY | <i>Number</i> | The number of signatures to authorize. |
| documents/document_id | OPTIONAL | <i>String</i> | It is Ascertia's custom parameter that will represent the unique ID for the document. |
| documents/document_name | OPTIONAL | <i>String</i> | It is Ascertia's custom parameter that will represent the name of the document. |
| hash | CONDITIONAL | <i>String</i> | One or more Base64-encoded hash values to be signed. It allows the server to bind the SAD to the hash(es), thus preventing an authorization to be used to sign a different content. If the SCAL parameter returned by credentials/info method, for the current credentialID is "2", the hash parameter shall be used and the number of hash values should correspond to the value in numSignatures. If the SCAL parameter is "1", the hash parameter is OPTIONAL. |
| PIN | CONDITIONAL | <i>String</i> | The PIN provided by the user. It shall be used only when <i>authMode</i> from credentials/info is "explicit" and PIN/presence is not "false". |
| OTP | CONDITIONAL | <i>String</i> | The OTP provided by the user. It shall be used only when <i>authMode</i> from credentials/info method is "explicit" and OTP/presence is not "false". |
| description | OPTIONAL | <i>String</i> | A free form description of the authorization transaction in the <i>lang</i> language. The maximum size of the string is 5000 characters. It can be useful when <i>authMode</i> from credentials/info method is "implicit" to provide some hints about the occurring transaction. |
| clientData | OPTIONAL | <i>String</i> | Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes. Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully. |

Response Parameters

| Parameters | Presence | Value | Description |
|------------|----------|-------|-------------|
|------------|----------|-------|-------------|

| | | | |
|-------------------|-------------|--------|---|
| SAD | MANDATORY | String | The Signature Activation Data (SAD) to be used as input to the signatures/signHash method. |
| expiresIn | OPTIONAL | Number | The lifetime in seconds of the SAD. If omitted, the default expiration time is 3600 (1 hour). |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.2.6 Credentials/extendTransaction

Extends the validity of a multi-signature transaction authorization by obtaining a new Signature Activation Data (SAD). This method SHALL be used in case of multi-signature transaction when the API method `signatures/signHash` is invoked multiple times with a single credential authorization event.

| | | |
|---|---|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/credentials/extendTransaction">https://<server>:8778/adss/service/ras/csc/v1/credentials/extendTransaction | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| Request Body | <pre>{ "credentialID": "JohnDoe", "SAD": "_TiHRG-bAH3XIFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw", "hash": ["sTOgwOm+474gFj0q0x1iSNspKqbcse4leiqDg/HWul="] }</pre> | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "signatures": ["KeTob5gl26S2tmXjqN...MRGtoew=="] }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter SAD" }</pre> |
| 400 | Bad Request | { |

| | | |
|-----|-------------|--|
| | | <pre>"error": "invalid_request", "error_description": "Invalid parameter SAD" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter credentialID " }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Invalid parameter credentialID" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) array parameter hash" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Invalid Base64 hash string parameter" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Invalid digest value length" }</pre> |

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-------------------------|----------------------------|--|
| credentialID | REQUIRED | <i>String</i> | The unique identifier associated to the credential. |
| hash | REQUIRED Conditional | <i>Array of String</i> | One or more Base64-encoded hash values to be signed. It allows the server to bind the new SAD to the hash, thus preventing an authorization to be used to sign a different content. It SHALL be used if the SCAL parameter returned by credentials/info for the current <i>credentialID</i> is "2", otherwise it is OPTIONAL. |
| SAD | REQUIRED | <i>String</i> | The current unexpired Signature Activation Data. This token is returned by the credentials/authorize or by the previous call to credentials/extendTransaction . |
| clientData | OPTIONAL | <i>String</i> | Arbitrary data from the signature application. It can be used to handle a transaction identifier or |

| | | |
|--|--|--|
| | | <p>other application-specific data that may be useful for debugging purposes.</p> <p>Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully.</p> |
|--|--|--|

Response Parameters

| Parameters | Presence | Value | Description |
|------------|----------|---------------|--|
| SAD | REQUIRED | <i>String</i> | The new Signature Activation Data required to sign multiple times with a single authorization. |
| expiresIn | OPTIONAL | <i>Number</i> | The lifetime in seconds of the SAD. If omitted, the default expiration time is 3600 (1 hour). |

3.2.7 Credentials/sendOTP

Start an online One-Time Password (OTP) generation mechanism associated with a credential and managed by the remote service. This will generate a dynamic one-time password that will be delivered to the user who owns the credential through an agreed communication channel managed by the remote service (e.g. SMS, email, app, etc.). This method SHOULD only be used with “online” OTP generators.

| | | |
|---|--|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/credentials/sendOTP">https://<server>:8778/adss/service/ras/csc/v1/credentials/sendOTP | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| Request Body | <pre>{ "credentialID": "JohnDoe", "clientData": "12345678" }</pre> | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter credentialID " }</pre> |
| 400 | Bad Request | <pre>{</pre> |

| | | |
|--|--|--|
| | | <pre>"error": "invalid_request", "error_description": "Invalid parameter credentialID" }</pre> |
|--|--|--|

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|----------|--------|---|
| credentialID | REQUIRED | String | The unique identifier associated to the credential. |
| clientData | OPTIONAL | String | Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes. Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully. |

Response Parameters

This method has no output values and the response returns “No Content” status.

3.2.8 Signatures/signHash

Calculate the remote digital signature of one or multiple hash values provided as an input. This method requires providing credential authorization in the form of [Signature Activation Data \(SAD\)](#).

| | |
|---|--|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/signatures/signHash">https://<server>:8778/adss/service/ras/csc/v1/signatures/signHash | |
| HTTP Verb | POST |
| Content-Type | application/json |
| Accept | application/json |
| Authorization | Bearer {access_token} |
| Request Body | <pre>{ "credentialID": "JohnDoe", "SAD": "_TiHRG-bAH3XIFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw", "documents": [{ "document_id": 123, "document_name": "Document Name 123", }, { "document_id": 456, "document_name": "Document Name 456", },], "hash": ["sTOgwOm+474gFj0q0x1iSNspKqbcse4leiQDg/HWul="], "hashAlgo": "2.16.840.1.101.3.4.2.1", }</pre> |

| | <pre>"signAlgo": "1.2.840.113549.1.1.1" }</pre> <p>Note: The 'documents', 'document_id' and 'document_name' are Ascertia's custom parameters and are optional in this case.</p> | |
|-------------|--|---|
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "signatures": ["KeTob5gl26S2tmXjqN...MRGtoew=="] }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter SAD" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Invalid parameter SAD" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter credentialID " }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Invalid parameter credentialID" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Missing (or invalid type) array parameter hash" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "Empty hash array" }</pre> |

| | | |
|-----|-------------|---|
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid Base64 hash string parameter" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter signAlgo" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter hashAlgo" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid parameter hashAlgo" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid parameter signAlgo" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid digest value length" } |
| 400 | Bad Request | { "error": "invalid_otp", "error_description": "The OTP is invalid" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Signing certificate 'O=[organization],CN=[common_name]' is expired." } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "Invalid parameter clientData " } |

Table 6 – Signatures/SignHash

Request Parameters

| Parameters | Presence | Value | Description |
|-------------------------|-------------|---------------|--|
| credentialID | MANDATORY | <i>String</i> | The unique identifier associated to the credential. |
| SAD | MANDATORY | <i>String</i> | The Signature Activation Data returned by the Credential Authorization methods. |
| documents/document_id | OPTIONAL | <i>String</i> | It is Ascertia's custom parameter that will represent the unique ID for the document. |
| documents/document_name | OPTIONAL | <i>String</i> | It is Ascertia's custom parameter that will represent the name of the document. |
| hash | MANDATORY | <i>String</i> | One or more hash values to be signed. This parameter shall contain the Base64-encoded raw message digest(s). |
| hashAlgo | CONDITIONAL | <i>String</i> | The OID of the algorithm used to calculate the hash value(s). This parameter shall be omitted or ignored if the hash algorithm is implicitly specified by the <i>signAlgo</i> algorithm. Only hashing algorithms as strong or stronger than SHA256 shall be used. The hash algorithm should follow the recommendations of ETSI TS 119 312. |
| signAlgo | MANDATORY | <i>String</i> | The OID of the algorithm to use for signing. It shall be one of the values allowed by the credential as returned in <i>keyAlgo</i> by the credentials/info method. |
| signAlgoParams | CONDITIONAL | <i>String</i> | The Base64-encoded DER-encoded ASN.1 signature parameters, if required by the signature algorithm. Some algorithms like RSASSA-PSS, as defined in RFC 8917, may require additional parameters. |
| clientData | OPTIONAL | <i>String</i> | Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes. Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully. |

Response Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|---------------|---|
| signatures | MANDATORY | <i>String</i> | One or more Base64-encoded signed hash(es). In case of multiple signatures, the signed hash(es) shall be returned in the same order as the corresponding hashes provided as an input parameter. |

| | | | |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.2.9 Application Meta Information

This call returns the meta information and the list of endpoints implemented by the service.

| <a href="https://<server>:8778/adss/service/ras/csc/v1/info">https://<server>:8778/adss/service/ras/csc/v1/info | | |
|---|------------------|--|
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | No Auth | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "specs": "1.0.3.0", "name": "Ascertia RAS", "logo": "https://localhost:8777/images/logo.png", "region": "GB", "lang": "en-gb", "description": "RAS - CSC Service provides remote authorization service implementing protection profiles", "oauth2BaseURI": "http://localhost:8777/adss/service/ras/csc/v1", "authType": ["basic", "oauth2code", "oauth2client"], "methods": ["auth/login", "auth/revoke", "credentials/list", "credentials/info", "credentials/authorize", "signatures/signHash", "oauth2/authorize", "oauth2/token", "oauth2/revoke"], }</pre> |
| 404 | Not Found | HTTP Status 404 – Not Found |

Table 7 – Application Meta Information

Request Parameters

| Parameters | Presence | Value | Description |
|------------|----------|--------|--|
| lang | OPTIONAL | String | <p>Request a preferred language of the response to the remote service, specified according to RFC 5646.</p> <p>If present, the remote service shall provide language-specific responses using the specified language. If the specified language is not supported then it shall provide these responses in the language as specified in the <i>lang</i> output parameter.</p> |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------|-------------|--------|--|
| specs | MANDATORY | String | <p>The version of the specification implemented by the RAS Service. The format of the string is Major.Minor.x.y, where Major is a number equivalent to the API version (e.g. 1 for API v1) and Minor is a number identifying the version update, while x and y are subversion numbers. The value corresponding to implemented specification is "1.0.3.0".</p> |
| name | MANDATORY | String | <p>The commercial name of the remote service. The maximum size of the string is 255 characters.</p> |
| logo | MANDATORY | String | <p>The URI of the image file containing the logo of the RAS Service which shall be published online. The image shall be in either JPEG or PNG format and not larger than 256x256 pixels.</p> |
| region | MANDATORY | String | <p>The ISO 3166-1 Alpha-2 code of the Country where the RAS Service is established (e.g. ES for Spain).</p> |
| lang | MANDATORY | String | <p>The language used in the responses, specified according to RFC 5646.</p> |
| description | MANDATORY | String | <p>A free form description of the RAS Service in the lang language. The maximum size of the string is 255 characters.</p> |
| oauth2 | CONDITIONAL | String | <p>The base URI of the OAuth 2.0 authorization server endpoint supported by the remote service for service authorization and/or credential authorization. The parameter shall be present in any of the following cases:</p> <ul style="list-style-type: none"> The authType parameter contains "oauth2code" or "oauth2client"; The remote service supports the value "oauth2code" for the authMode parameter returned by credentials/info. <p>This URI shall be combined with the OAuth 2.0 endpoints.</p> |

| | | | |
|----------|-----------|--------|--|
| authType | MANDATORY | String | <p>One or more values corresponding to the service authorization mechanisms supported by the remote service to authorize the access to the API. The following mechanisms are supported in RAS Service:</p> <ul style="list-style-type: none"> • Basic: In case of HTTP Basic Authentication. • OAuth2code: In case of OAuth 2.0 with authorization code flow. • OAuth2client: In case of OAuth 2.0 with client credentials flow. |
| methods | MANDATORY | String | <p>The list of names of all the API methods described in the specification that are implemented and supported by the RAS Service.</p> |

3.2.10 OAuth2/Authorize

It does not specify a regular CSC API method, but rather the URI path component of the address of the web page allowing the user to sign-in to the remote service to authorize the signature application or to authorize a credential. The complete URL to invoke the OAuth 2.0 authorization server is obtained by adding `oauth2/authorize` to the base URI of the authorization server as returned in the `oauth2` parameter by the "info" method and it does not necessarily include the base URI of the remote service API.

| | |
|---|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/oauth2/authorize">https://<server>:8778/adss/service/ras/csc/v1/oauth2/authorize | |
| HTTP Verb | GET |
| Content-Type | |
| Accept | |
| Parameters | <pre>//Service Authorization response_type=code& client_id=samples_test_client& redirect_uri=http://localhost:8777& scope=service& lang=en-UK& state=123456& profile_id=adss:ras:profile:001 // Credentials Authorization response_type=code& client_id=samples_test_client& redirect_uri=http://localhost:8777& scope=credential& credentialID=sample-key& numSignatures=2& hash= MTIzNDU2Nzg5MHF3ZXJ0enVpb3Bhc2RmZ2hqa2zDtnI4& state=12345</pre> |
| | |

| Status Code | Message | Response Body |
|-------------|-------------|---|
| 302 | Found | Location: <OAuth2_redirect_uri> ? code=12234&state=121212 |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |

Table 8 – OAuth2/Authorize

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|--|
| response_type | MANDATORY | String | The value shall be “code”. |
| client_id | MANDATORY | String | The unique “client ID” previously assigned to the signature application by the RAS Service. |
| redirect_uri | OPTIONAL | String | The URL where the user will be redirected after the authorization process has completed. Only a valid URI pre-registered with the RAS Service shall be passed. If omitted, the service will use the default redirect URI pre-registered by the signature application. |
| scope | OPTIONAL | String | <p>The scopes of the access request are mentioned below:</p> <ul style="list-style-type: none"> • Service: It shall be used to obtain an authorization code suitable for service authorization. • Credential: It shall be used to obtain an authorization code suitable for credentials authorization. <p>The parameter is OPTIONAL. The defaults scope is “service” in case it is omitted.</p> |
| lang | OPTIONAL | String | <p>Request a preferred language according to RFC 5646.</p> <p>If specified, the authorization server should render the authorization web page in this language, if supported. If omitted and an Accept-Language header is passed, the authorization server should render the authorization web page in the language declared by the header value, if supported.</p> <p>The authorization server shall render the web page in its own preferred language otherwise.</p> |
| state | OPTIONAL | String | Up to 255 bytes of arbitrary data from the signature application that will be passed back to the redirect URI. The use is recommended for preventing cross-site request forgery. |

| | | | |
|---------------|-------------|--------|---|
| profile_id | OPTIONAL | String | It is Ascertia's custom parameter that will represent the RAS profile ID being used. |
| credentialID | CONDITIONAL | String | The identifier associated to the credential to authorize. It shall be used only if the scope of the OAuth 2.0 authorization request is "credential". This parameter value may contain characters that are reserved, unsafe or forbidden in URLs and therefore shall be url-encoded by the signature application. |
| numSignatures | CONDITIONAL | Number | The number of signatures to authorize. Multi-signature transactions can be obtained by using a combination of array of hash values and by calling multiple times the signatures/signHash method. It shall be used only if the scope of the OAuth 2.0 authorization request is "credential". |
| hash | CONDITIONAL | String | <p>One or more base64url-encoded hash values to be signed. It allows the server to bind the SAD to the hash, thus preventing an authorization to be used to sign a different content. It shall be used if the SCAL parameter returned by credentials/info method, for the current credentialID is "2", otherwise it is OPTIONAL. Multiple hash values can be passed as comma separated values, e.g. oauth2/authorize?hash=dnN3ZX...ZmRm,ZjlxM3...Z2Zk,...</p> <p>The order of multiple values does not have to match the order of hashes passed to signatures/signHash method.</p> |
| description | OPTIONAL | String | A free form description of the authorization transaction in the lang language. The maximum size of the string is 5000 characters. It can be useful to provide some hints about the occurring transaction. |
| account_token | OPTIONAL | String | An account_token may be required by a RSSP if their authorization server has a restricted access. The value is a JSON Web Token (JWT) according to RFC 7519. |
| clientData | OPTIONAL | String | <p>Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes.</p> <p>Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully.</p> |

Response Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| code | MANDATORY | String | The authorization code generated by the authorization server. It shall be bound to the client identifier and the redirection URI. It shall |

| | | | |
|-------------------|-------------|--------|---|
| | | | expire shortly after it is issued to mitigate the risk of leaks. The signature application cannot use the value more than once. |
| state | CONDITIONAL | String | Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes. Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |
| error_uri | OPTIONAL | String | A URI identifying a human-readable web page with information about the error. It may be returned only in case of an error. |

3.2.11 OAuth2/Token – Authorization Code Flow

Obtain an OAuth 2.0 bearer access token from the authorization server by passing the authorization code or refresh token returned by the authorization server after a successful user authentication, along with the client ID and client secret in possession of the signature application.

| | | |
|---|---|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/oauth2/token">https://<server>:8778/adss/service/ras/csc/v1/oauth2/token | | |
| HTTP Verb | POST | |
| Content-Type | application/x-www-form-urlencoded | |
| Accept | application/json | |
| Request Header | | |
| profile_id | adss:ras:profile:001 | |
| Request Body | <pre>grant_type=authorization_code& client_id=samples_test_client& client_secret=jr67gj0h76gr83nf8734nj59g4he895jh87nr& code=ssd34343& redirect_uri=http://localhost:8777</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>// Service Authorisation Response { "access_token":"KeTob5gl26S2tmXjqN...MRGtoew==" "refresh_token":"KeTob5gl26S2tmXjqN...MRGtoew=="</pre> |

| | | |
|-----|-------------|---|
| | | <pre> "token_type": "Bearer" "expires_in": "3600" } // Credentials Authorisation Response { "access_token": "KeTob5gl26S2tmXjqN...MRGtoew==" "token_type": "SAD" "expires_in": "3600" } </pre> |
| 400 | Bad Request | <pre> "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." } </pre> |

Table 9 – OAuth2/token

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-------------|--------|--|
| grant_type | MANDATORY | String | The grant type, which depends on the type of OAuth 2.0 flow: <ul style="list-style-type: none"> authorization_code: It shall be used in case of Authorization Code Grant. client_credentials: It shall be used in case of Client Credentials Grant. refresh_token: It shall be used in case of Refresh Token flow. |
| refresh_token | CONDITIONAL | String | The long-lived refresh token returned from the previous session. This shall be used only when the scope of the OAuth 2.0 authorization request is "service" and <i>grant_type</i> is "refresh_token" to reauthenticate the user according to the method described in RFC 6749. |
| client_id | MANDATORY | String | The unique "client ID" previously assigned to the signature application by the remote service. |
| client_secret | CONDITIONAL | String | This is the "client secret" previously assigned to the signature application by the remote service. It shall be passed if no authorization header and no client assertion is used. |
| code | CONDITIONAL | String | The authorization code returned by the authorization server. It shall be bound to the client identifier and the redirection URI. This shall be used only when <i>grant_type</i> is "authorization_code". |

| | | | |
|-----------------------|-------------|--------|--|
| client_assertion | CONDITIONAL | String | The assertion being used to authenticate the client. Specific serialization of the assertion is defined by profile documents. It shall be passed if no authorization header and no <i>client_secret</i> is used. |
| client_assertion_type | CONDITIONAL | String | The format of the assertion as defined by the authorization server. The value will be an absolute URI. It shall be passed if a client assertion is used. |
| redirect_uri | CONDITIONAL | String | The URL where the user was redirected after the authorization process completed. It is used to validate that it matches the original value previously passed to the authorization server. This shall be used only if the <i>redirect_uri</i> parameter was included in the authorization request, and their values shall be identical. |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|--|
| access_token | MANDATORY | String | The short-lived access token to be used depending on the scope of the OAuth 2.0 authorization request. When the scope is “service” then the authorization server returns a bearer token to be used as the value of the “Authorization: Bearer” in the HTTP header of the subsequent API requests within the same session. When the scope is “credential” then the authorization server returns a Signature Activation Data token to authorize the signature request. This value should be used as the value for the SAD parameter when invoking the signatures/signHash method. |
| refresh_token | OPTIONAL | String | The long-lived refresh token used to re-authenticate the user on the subsequent session based on the method described in RFC 6749. The presence of this parameter is controlled by the user and is allowed only when the scope of the OAuth 2.0 authorization request is “service”. In case <i>grant_type</i> is “refresh_token” the authorization server may issue a new refresh token, in which case the client shall discard the old refresh token and replace it with the new refresh token. |

| | | | |
|-------------------|-------------|--------|---|
| token_type | MANDATORY | String | When the scope is "service", this specifies a "Bearer" token type as defined in RFC6750 . When the scope is "credential", this specifies a "SAD" token type. |
| expires_in | OPTIONAL | Number | The lifetime in seconds of the service access token. If omitted, the default expiration time is 3600 sec. (1 hour). |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.2.12 OAuth2/Token – Client Credentials Flow

Obtain an OAuth 2.0 bearer access token from the authorization server by passing the client credentials which is pre-assigned by the authorization server to the signature application along with the client ID and client secret in possession of the signature application.

| | | |
|---|---|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/oauth2/token">https://<server>:8778/adss/service/ras/csc/v1/oauth2/token | | |
| HTTP Verb | POST | |
| Content-Type | application/x-www-form-urlencoded | |
| Accept | application/json | |
| Request Headers | | |
| profile_id | adss:ras:profile:001 | |
| Request Body | grant_type=client_credentials& client_id=samples_test_client& client_secret=jr67gj0h76gr83nf8734nj59g4he895jh87nr | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | // Service Authorization Response { "access_token":"KeTob5gl26S2tmXjqN...MRGtoew==" "refresh_token":"KeTob5gl26S2tmXjqN...MRGtoew==" "token_type":"Bearer" "expires_in":"3600" } |
| 400 | Bad Request | { "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." } |

Table 10 – OAuth2/token

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-------------|---------------|--|
| grant_type | MANDATORY | <i>String</i> | The grant type, which depends on the type of OAuth 2.0 flow. In this case it will be "client_credentials". |
| client_id | MANDATORY | <i>String</i> | The unique "client ID" previously assigned to the signature application by the remote service. |
| client_secret | CONDITIONAL | <i>String</i> | This is the "client secret" previously assigned to the signature application by the remote service. It shall be passed if no authorization header and no client assertion is used. |
| profile_id | OPTIONAL | <i>String</i> | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------------|---|
| access_token | MANDATORY | <i>String</i> | The short-lived access token to be used depending on the scope of the OAuth 2.0 authorization request. |
| token_type | MANDATORY | <i>String</i> | In case of client_credentials this will be "bearer". |
| expires_in | OPTIONAL | <i>Number</i> | The lifetime in seconds of the service access token. If omitted, the default expiration time is 3600 sec. (1 hour). |
| error_code | CONDITIONAL | <i>String</i> | The error code. |
| error_description | CONDITIONAL | <i>String</i> | Error description message. |

3.2.13 OAuth2/Token – Refresh Token Flow

Obtain an OAuth 2.0 bearer access token from the authorization server by passing the client credentials with refresh token which is pre-assigned by the authorization server to the signature application along with the client ID and client secret in possession of the signature application.

| | |
|---|-----------------------------------|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/oauth2/token">https://<server>:8778/adss/service/ras/csc/v1/oauth2/token | |
| HTTP Verb | POST |
| Content-Type | application/x-www-form-urlencoded |
| Accept | application/json |
| Request Headers | |
| profile_id | adss:ras:profile:001 |

| | | |
|--------------------|---|---|
| Request Body | <pre>grant_type=refresh_token& refresh_token=Base64& client_id=samples_test_client& client_secret=jr67gj0h76gr83nf8734nj59g4he895jh87nr</pre> | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>// Service Authorization Response { "access_token":"KeTob5gl26S2tmXjqN...MRGtoew==" "refresh_token":"KeTob5gl26S2tmXjqN...MRGtoew==" "token_type":"Bearer" "expires_in":"3600" }</pre> |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |

Table 10 – OAuth2/token

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-------------|--------|--|
| grant_type | MANDATORY | String | The grant type, which depends on the type of OAuth 2.0 flow. In this case it will be "refresh_token". |
| client_id | MANDATORY | String | The unique "client ID" previously assigned to the signature application by the remote service. |
| client_secret | CONDITIONAL | String | This is the "client secret" previously assigned to the signature application by the remote service. It shall be passed if no authorization header and no client assertion is used. |
| refresh_token | MANDATORY | String | Existing refresh token used to get the new access and refresh tokens. |
| profile_id | OPTIONAL | String | RAS Profile ID that will be used to process the request (max. 50 characters). |

Response Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|--|
| access_token | MANDATORY | String | The short-lived access token to be used depending on the scope of the OAuth 2.0 authorization request. |

| | | | |
|-------------------|-------------|--------|---|
| refresh_token | OPTIONAL | String | <p>The long-lived refresh token used to re-authenticate the user on the subsequent session based on the method described in RFC 6749.</p> <p>The presence of this parameter is controlled by the user and is allowed only when the scope of the OAuth 2.0 authorization request is "service".</p> <p>In case <i>grant_type</i> is "refresh_token" the authorization server may issue a new refresh token, in which case the client shall discard the old refresh token and replace it with the new refresh token.</p> |
| token_type | MANDATORY | String | In case of refresh_token this will be "bearer". |
| expires_in | OPTIONAL | Number | The lifetime in seconds of the service access token. If omitted, the default expiration time is 3600 sec. (1 hour). |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

3.2.14 OAuth2/Revoke

Revoke an access token or refresh token that was obtained from the authorization server, as described in RFC 7009. This method may be used to enforce the security of the remote service. When the signature application needs to terminate a session, it is RECOMMENDED to invoke this method to prevent further access by reusing the token.

| | | |
|---|---|---|
| <a href="https://<server>:8778/adss/service/ras/csc/v1/oauth2/revoke">https://<server>:8778/adss/service/ras/csc/v1/oauth2/revoke | | |
| HTTP Verb | POST | |
| Content-Type | application/x-www-form-urlencoded | |
| Accept | application/json | |
| Request Body | <pre>token= jr67gj0h76gr83nf8734nj59g4he895jh87nr& token_type_hint=access_token/refresh_token& client_id=samples_test_client& client_secret=jr67gj0h76gr83nf8734nj59g4he895jh87nr</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad Request | <pre>{ "error": "invalid_request", "error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }</pre> |

Table 11 – OAuth2/revoke

Request Parameters

| Parameters | Presence | Value | Description |
|-----------------------|-------------|---------------|---|
| token | MANDATORY | <i>String</i> | The token that the signature application wants to get revoked. |
| token_type_hint | OPTIONAL | <i>String</i> | Specifies an optional hint about the type of the token submitted for revocation. If the parameter is omitted, the authorization server should try to identify the token across all the available tokens. |
| client_id | CONDITIONAL | <i>String</i> | The unique “client ID” previously assigned to the signature application by the remote service. It shall be passed if no authorization header is used. |
| client_secret | CONDITIONAL | <i>String</i> | This is the “client secret” previously assigned to the signature application by the remote service. It shall be passed if no authorization header and no client assertion is used. |
| client_assertion | CONDITIONAL | <i>String</i> | The assertion being used to authenticate the client. Specific serialization of the assertion is defined by profile documents. It shall be passed if no authorization header and no <i>client_secret</i> is used. |
| client_assertion_type | CONDITIONAL | <i>String</i> | The format of the assertion as defined by the authorization server. The value will be an absolute URI. It shall be passed if a client assertion is used. |
| clientData | OPTIONAL | <i>String</i> | Arbitrary data from the signature application. It can be used to handle a transaction identifier or other application-specific data that may be useful for debugging purposes. Warning: This parameter may expose sensitive data to the remote service. Therefore, it should be used carefully. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------------|----------------------------|
| error_code | CONDITIONAL | <i>String</i> | The error code. |
| error_description | CONDITIONAL | <i>String</i> | Error description message. |

4 Mobile Application Interfaces

A mobile app must interact with ADSS RAS to handle these services:

- Registration of the user's mobile device for remote authorisation
- Allowing the user to receive, authorise and send remote signing requests/responses

Mobile apps integrate with ADSS RAS Service using RESTful APIs. This section details each API method.

4.1 Authenticate Client

This API is used to authenticate a client using its credentials. The RAS Service returns an access token on successful authentication of the client.

| | | |
|---|---|---|
| <a href="https://<server>:8778/adss/service/ras/v1/authenticate">https://<server>:8778/adss/service/ras/v1/authenticate | | |
| HTTP Verb | POST | |
| Content-Type | application/x-www-form-urlencoded | |
| Accept | application/json | |
| Request Body | <code>client_id=samples_test_client & client_secret=121212 & grant_type=client_credentials</code> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | { "access_token":"2YotnFZFEjr1zCsicMWpAA", "expires_in":3600 } |
| 400 | Bad Request | For Error information in client credentials request refer OAuth RFC 6749 at: https://tools.ietf.org/html/rfc6749#section-5.2 |

Table 1 – Authenticate Application

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|---|
| client_id | MANDATORY | String | Client ID registered in ADSS Client Manager |
| client_secret | MANDATORY | String | Client Secret generated against the client in ADSS Client Manager |
| grant_type | MANDATORY | String | Grant type would be the client_credentials |

Response Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|--|
| access_token | MANDATORY | String | It will return the client access token after the authentication of client ID and secret. |
| expires_in | MANDATORY | String | Token expiry mentioned in the seconds. |

4.2 Authenticate User

This call initiates the user authentication on the mobile application. A user can be authenticated using the following authentication methods. These methods can be configured in the RAS Profile:

- Authenticate user with OTP(s) (Either SMS or Email or Both SMS/Email)
- Authenticate user with QR Code
- No Authentication

Authenticate user with OTP(s):

If this option is enabled, it means user will be authenticated using the OTPs. RAS will send a request to SAM to generate either a single or two OTPs according to the option "SMS OTP" and "Email OTP" selected in the RAS Profile. The SAM will generate the OTP(s) and return to RAS that will send the OTP(s) to user's mobile number or email. RAS will return the authentication type "OTP" in response to mobile application to let it know that the user will be authenticated using OTP(s) that are sent to his/her mobile/email. The mobile application should display fields to user to enter the OTP(s) and once user enters the OTP, the mobile application will invoke another RAS API (Verify OTPs) to verify these OTP(s). The API is discussed in a later section.

Authenticate user with QR Code:

If this option will be selected in RAS Profile, the RAS Service will instantly return the response to mobile application with authentication type "qrCode". This will be an indication that the user will be authenticated using a QR Code so the mobile app will ask the user to go to QR code page and scan the QR code. Once the mobile app scans the QR Code, it will send this to RAS for verification by calling another API (Verify QR Code).

No Authentication:

In this case, the RAS will just verify the user a registered one in the SAM Service. After getting confirmation from SAM the RAS will generate the access and refresh tokens for this user and return to client application. The presence of access token in response will be an indication for mobile app that the user has been authenticated.

Note: The clients should use "No Authentication" option in scenarios where the mobile app has its own mechanism of user authentication. The mobile app will authenticate the user and then request for an access token from RAS to access its APIs.

| | | |
|---|---|--|
| <a href="https://<server>:8778/adss/service/ras/v1/user/enrol">https://<server>:8778/adss/service/ras/v1/user/enrol | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {application_access_token} | |
| Request Body | <pre>{ "user_id": "John_Doe", }</pre> | |
| Response Header | | |
| authentication_methods | true | |
| Status Code | Message | Response Body |
| 200 | OK | <p>If OTP Authentication is configured in RAS Profile.</p> <p>If both SMS and Email OTPs are sent to user:</p> |

| | | |
|-----|----|--|
| | | <pre>{ "auth_type": "OTP", "otp_info": [{ "otp_type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }, { "otp_type": "SMS_OTP", "sent_to": "+448007720442" }] }</pre> |
| 200 | OK | <p>If one OTP will be sent on user email:</p> <pre>{ "auth_type": "OTP", "otp_info": [{ "otp_type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }] }</pre> |
| 200 | OK | <p>If one OTP will be sent to user's mobile:</p> <pre>{ "auth_type": "OTP", "otp_info": [{ "otp_type": "SMS_OTP", "sent_to": "+448007720442" }] }</pre> |
| 200 | OK | <p>If QR Code authentication is configured:</p> <pre>{ "auth_type": "QR_CODE", }</pre> |
| 200 | OK | <p>If no authentication is configured:</p> <pre>{ "auth_type": "NO_AUTHENTICATION", "token_info": { "access_token": "eyJhbGciOiJIUzI1diIud...96RDo", "refresh_token": "eyJhbGciOiJIUzI1diIud...ymjGp-E", } }</pre> |

| | | |
|-----|--|---|
| | | <pre>"token_type": "bearer", "expires_in": 3600 } }</pre> |
| 400 | | |
| 500 | | |

Table 2 – Authenticate User

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|---|
| user_id | MANDATORY | String | User ID as registered by the business application in ADSS Server SAM. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|--|
| auth_type | MANDATORY | String | <p>Authentication type configured in RAS Profile. The following values can be found in this parameter:</p> <ul style="list-style-type: none"> • OTP • QR_CODE • NO_AUTHENTICATION <p>Note: The values OTP, QR_CODE and NO_AUTHENTICATION are case-sensitive.</p> |
| otp_info | CONDITIONAL | String | Contains information related to the types of OTPs (Email/SMS) and the mobile number and email of the user. It applies when OTP authentication is enabled in RAS Profile. |
| token_info | CONDITIONAL | String | Contains the OAuth access & refresh tokens and the expiry. It applies when “No Authentication” option is enabled in RAS Profile. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

4.3 Verify OTPs

If the OTP authentication will be enabled in RAS Service, the user will receive either one or two OTPs on his mobile number or email. The user will provide these OTPs to this API. After successful OTPs verification, access and refresh tokens are returned.

| | |
|---|------|
| https://server:8778/adss/service/ras/v1/authentication/otp/verify | |
| HTTP Verb | POST |

| | | |
|--------------------|--|---|
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {application_access_token} | |
| Request Body | <p>If user had received two OTPs:</p> <pre>{ "user_id": "User ID", "otp_info": [{ "otp": "258456987", "otp_type": "SMS_OTP" }],{ "otp": "258456987", "otp_type": "EMAIL_OTP" }] }</pre> <p>If user had received a single OTP on mobile:</p> <pre>{ "user_id": "User ID", "otp_info": [{ "otp": "258456987", "otp_type": "SMS_OTP" }] }</pre> <p>If user had received one OTP via email:</p> <pre>{ "user_id": "User ID", "otp_info": [{ "otp": "258456987", "otp_type": "EMAIL_OTP" }] }</pre> | |
| Status Code | Message | Response Body |
| 200 | Ok | <pre>{ "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWRXMTIzNDU2N285Lm9udG8iLCJ0eXBlIjoiYm9hcm91IiwiaWF0IjoiMTYxMjM0NTY3ODkifQ", "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWRXMTIzNDU2N285Lm9udG8iLCJ0eXBlIjoiYm9hcm91IiwiaWF0IjoiMTYxMjM0NTY3ODkifQ", "token_type": "bearer", "expires_in": 3600, }</pre> |
| 400 | Bad Request | |
| 401 | Unauthorized | |
| 403 | Forbidden | |

| | | |
|-----|-----------------------|--|
| 500 | Internal Server Error | |
|-----|-----------------------|--|

Table 3 – Verify OTP

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|-------------|--|
| user_id | MANDATORY | String | User ID as registered by the business application. |
| otp_info | MANDATORY | JSON Object | It contains the OTPs and their types i.e. SMS/Email. |
| otp | MANDATORY | String | OTP received by the user on his/her mobile/email. |
| otp_type | MANDATORY | String | Type of the OTP i.e. SMS or Email. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|---|
| access_token | MANDATORY | String | Access token of the user to use in subsequent API calls. |
| refresh_token | MANDATORY | String | Refresh token will be used to get the new access token without authenticating the user again. |
| expires_in | MANDATORY | String | It's token expiry mentioned in the seconds. |
| error | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

4.4 Renew Access Token

This API allows the renewal of an expired access token by providing the refresh token.

| | | |
|---|--|--|
| <a href="https://<server>:8778/adss/service/ras/v1/authenticate">https://<server>:8778/adss/service/ras/v1/authenticate | | |
| HTTP Verb | POST | |
| Content-Type | application/x-www-form-urlencoded | |
| Accept | application/json | |
| Request Body | grant_type=refresh_token&refresh_token=tGzv3JOkF0XG5Qx | |
| Status Code | Message | Response Body |
| 200 | OK | { "access_token": "2YotnFZFEjr1zCsicMWpAA", "refresh_token": "TRVfHTHcedfJGJFLGKKJ", |

| | | |
|-----|-------------|---|
| | | "expires_in":3600, } |
| 400 | Bad Request | For Error information in client credentials request refer OAuth RFC 6749 at: https://tools.ietf.org/html/rfc6749#section-5.2 |

Table 4 – Renew Access Token

Request Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|--|
| grant_type | MANDATORY | String | Grant type would be refresh_token. |
| refresh_token | MANDATORY | String | Refresh token which mobile application already received after user authentication. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|--|
| access_token | MANDATORY | String | New access token. |
| refresh_token | MANDATORY | String | New refresh token to cover in-activity time by the logged-in user. |
| expires_in | MANDATORY | String | Access token expiry in seconds. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error message description |

4.5 Device Registration

Once the mobile application gets the access token it can use other APIs of RAS. This API is used to register user's mobile device for remote signature authorisation purposes and request a certificate for the device's authorisation public key. Mobile application first needs to generate the key-pair in mobile device's software and hardware (Secure Enclave) and also generate the CSR (Certificate Signing Request). Once the CSR is generated, it will be sent in this API along with other information of the device. The RAS Service will return the certificate generated for the device after registering the device.

| | |
|---|---|
| <a href="https://<server>:8778/adss/service/ras/v1/authorization/certificate">https://<server>:8778/adss/service/ras/v1/authorization/certificate | |
| HTTP Verb | POST |
| Content-Type | application/json |
| Accept | application/json |
| Authorization | Bearer {access_token} |
| Request Body | { "csr": "MIICxDCCAawCAQAwfzELM[....]5f52oQ==", "device": { "device_id": " ASJMMN5389FF ", "device_name": "IPHONE X", |

| | <pre>"secure_element": true, "biometric": true, } }</pre> | |
|-------------|---|---|
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "alias": "hvcNAU+qCdXzADEA" "certificate": "MIItAYJKocNA[...]ZU+qCdXzADEA" }</pre> |
| 400 | Bad Request | |
| 500 | Internal Server Error | |

Table 5 – Request Device/User Certificate

Request Parameters

| Parameters | Presence | Value | Description |
|----------------|-----------|----------------|---|
| csr | MANDATORY | <i>String</i> | Base64 encoded value of the CSR. A certificate will be issued for mobile device against this CSR. |
| device_name | MANDATORY | <i>String</i> | Alias of the device. Later can be renamed. |
| device_id | MANDATORY | <i>String</i> | A unique device ID to identify the device, For example, UUID random number. |
| secure_element | MANDATORY | <i>Boolean</i> | Must set to “True” if device has a hardware Secure Element/Enclave. |
| biometric | MANDATORY | <i>Boolean</i> | Must set to “True” if device has biometric feature available. It can be TouchID, FaceID, Fingerprint etc. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|---------------|--|
| certificate | MANDATORY | <i>String</i> | Certificate generated for the device in base64 encoded format. |
| alias | MANDATORY | <i>String</i> | The certificate alias assigned by RAS Service to this certificate. |
| error_code | CONDITIONAL | <i>String</i> | The error code. |
| error_description | CONDITIONAL | <i>String</i> | A string with the description of the error_code. |

4.6 List Registered Devices

This API returns all the devices of a user that user has registered for use in remote authorised signing operations.

| | | |
|---|-----------------------------------|--|
| <a href="https://<server>:8778/adss/service/ras/v1/authorization/devices?user_id={user_id}">https://<server>:8778/adss/service/ras/v1/authorization/devices?user_id={user_id} | | |
| HTTP Verb | GET | |
| Accept | application/json | |
| Authorization | Bearer {application_access_token} | |
| Request Body | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>[{ "device_id": "id-001", "device_name": "iPhone", "secure_element": true, "biometric": true, }, { "device_id": "id-002", "device_name": "Samsung", "secure_element": true, "biometric": true, }]</pre> |
| 400 | Bad Request | |
| 500 | Internal Server Error | |

Table 7 – List Registered Devices

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| user_id | MANDATORY | String | User ID whose devices information needs to be retrieved. |

Response Parameters

| Parameters | Presence | Value | Description |
|----------------|-----------|--------|--|
| device_id | MANDATORY | String | Device ID. |
| device_name | MANDATORY | String | Device alias. |
| secure_element | MANDATORY | String | “True” if device has secure element/enclave. |

| | | | |
|-------------------|-------------|--------|---|
| biometric | MANDATORY | String | "True" if device has biometric feature available. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message |

4.7 Delete Device

This API deletes a user's device in RAS Service identified by {device_id}. A client application would use this interface to delete a user's device.

| | | |
|---|----------------------------|----------------------|
| https://server:8778/adss/service/ras/v1/authorization/devices/{device_id} | | |
| HTTP Verb | DELETE | |
| Accept | application/json | |
| Access Token | Bearer {user_access_token} | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 404 | Not Found | |
| 403 | Forbidden | |
| 500 | Internal Server Error | |

Table 7 - Delete Device

Request Parameters

| Parameters | Presence | Value | Description |
|-------------|-----------|--------|---|
| {device_id} | MANDATORY | String | Device ID which is already registered in ADSS Server. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

4.8 Get Pending Authorisation Request

This method returns a pending authorisation request. That is, where the business application has requested a signing operation that requires user authorisation.

| | |
|---|-----|
| <a href="https://<server>:8778/adss/service/ras/v1/authorization/request">https://<server>:8778/adss/service/ras/v1/authorization/request | |
| HTTP Verb | GET |

| | | |
|--------------------|-----------------------|--|
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "transaction_id": "932469001521668267", "request": "PEFDRj48Y2VydEFs[...]9BQ0Y+", "hash_algorithm": "SHA256" }</pre> |
| 400 | Bad Request | |
| 500 | Internal Server Error | |

Table 8 – Get Pending Signature Request

Response Parameters

| Parameters | Presence | Value | Description |
|----------------|-----------|--------|--|
| transaction_id | MANDATORY | String | Transaction ID that uniquely identifies the request |
| request | MANDATORY | String | <p>Pending authorisation request in base64 form - this is the “object” together with some additional data e.g. Device ID added and signed by the mobile app using the authorisation key. Value must be decoded before signing operation. After decoding the request, it will be look like this.</p> <pre><AuthorisationData> <OriginatorID>Virtual_CSP_Client</OriginatorID> <UserID>olcayatli@gmail.com</UserID> <CertificateID>416edc72-6c63-45aa-bb34a373102234df</CertificateID> <TransactionID>980551837300673581</TransactionID> <Salt>924552495291565632</Salt> <MetaData> <DisplayText>Data to be displayed</DisplayText> <DeviceID></DeviceID> </MetaData> <Documents> <Document id="b81e040a-a4d8-4134-92ff-2d4bf5e9116d"> <Name>b81e040a-a4d8-4134-92ff-2d4bf5e9116d</Name> <DigestValue>ypkP9L2tZ02JdfNr4X4X5SRur529uJqykdc5q5HDSiLNiYcLrys00S/H31yb8QZS&#xD;SOBYsFlVSj9/SKUqrhsUC5oEc/gr</DigestValue> </Document> </Documents></pre> |

| | | | |
|-------------------|-------------|--------|---|
| | | | <pre> <ValidityPeriod> <ValidFrom>2019-12-07T18:25:37</ValidFrom> <ValidTo>2019-12-07T18:42:17</ValidTo> </ValidityPeriod> <Signature> <DigestMethod>SHA256</DigestMethod> </AuthorisationData> </pre> |
| hash_algorithm | MANDATORY | String | Hash algorithm to be used for signing the authorized remote signing request. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

4.9 Authorise a Pending Request

This method authorises a pending request by sending the signed [Signature Activation Data \(SAD\)](#) against the pending authorisation request received as described above. That is, the value returned in section 4.8 above (together with some additional data) must be signed on the mobile device and returned using this API. The returned value must be base64 encoded. The hash algorithm is as returned in section 4.8 above, and the same value is returned here in the body request.

| | | |
|---|---|----------------------|
| <a href="https://<server>:8778/adss/service/ras/v1/authorization/request/{request_id}">https://<server>:8778/adss/service/ras/v1/authorization/request/{request_id} | | |
| HTTP Verb | PUT | |
| Authorization | Bearer {access_token} | |
| Content-Type | application/json | |
| Accept | application/json | |
| Request Body | <pre> { "request": "PEFDRj48Y2VydEFs[...]9BQ0Y+", "hash_algorithm": "SHA256" } </pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad request | |
| 500 | Internal Server Error | |

Table 9 – Confirm a Pending Signature Request

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|---|
| {request_id} | MANDATORY | String | Request ID received in section 4.8 above |
| Request | MANDATORY | String | Mobile app need to decode the request, add <DeviceID> and then sign with private key and add the signature in the |

| | | | |
|----------------|-----------|--------|--|
| | | | <p>request. This will be sent in request in Base64 encoded format. This signed request also called the SAD (Signature Activation Data) will look like this:</p> <pre> <AuthorisationData> <OriginatorID>Virtual_CSP_Client</OriginatorID > <UserID>olcayatli@gmail.com</UserID> <CertificateID>416edc72-6c63-45aa- bb34a373102234df</CertificateID> <TransactionID>980551837300673581</Transaction ID> <Salt>924552495291565632</Salt> <MetaData> <DisplayText>Data to be displayed</DisplayText> <DeviceID>ad1e60a6-5e23-4b52-a127- 27f41c224c05</DeviceID> </MetaData> <Documents> <Document id="b81e040a-a4d8-4134-92ff- 2d4bf5e9116d"> <Name>b81e040a-a4d8-4134-92ff- 2d4bf5e9116d</Name> <DigestValue>ypkP9L2tZ02JdfNr4X4X5SRur529uJqyk dc5q5HDSiLNiYcLrys00S/H31yb8QZS&#xD;SOBYsF1Vsj 9/SKUqrhsUC5oEc/gr</DigestValue> </Document> </Documents> <ValidityPeriod> <ValidFrom>2019-12-07T18:25:37</ValidFrom> <ValidTo>2019-12-07T18:42:17</ValidTo> </ValidityPeriod> <Signature> <DigestMethod>SHA256</DigestMethod> <SignatureValue>MEUCID/kiJWAIqzwOp/hi+FUbJwsjd csEoBNw1IF8sXA8XbDAiEAKGgQomyoJ2iR0ra9KGBFW/zX i6tbsn5M49YiaPnc+L8=</SignatureValue> </AuthorisationData> </pre> |
| hash_algorithm | MANDATORY | String | Hashing algorithm used for signing SAD. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

4.10 Cancel a Pending Authorisation Request

This method cancels a pending authorisation request. That is, the user decides to decline the authorisation request sent to the mobile device.

| | | |
|---|-----------------------|----------------------|
| <a href="https://<server>:8778/adss/service/ras/v1/authorization/request/{request_id}">https://<server>:8778/adss/service/ras/v1/authorization/request/{request_id} | | |
| HTTP Verb | DELETE | |
| Authorization | Bearer {access_token} | |
| Accept | application/json | |
| Request Body | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 400 | Bad request | |
| 500 | Internal Server Error | |

Table 10 – Cancel a Pending Signature Request

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|--|
| {request_id} | MANDATORY | String | Request ID for which user cancelled the authorization. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

4.11 Users Profile

This API is used to get user's profile information from ADSS.

| | | |
|---|-----------------------|--------------------------|
| <a href="https://<server>:8778/adss/service/ras/v1/users/profile">https://<server>:8778/adss/service/ras/v1/users/profile | | |
| HTTP Verb | GET | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | { "user_id": "Alice", |

| | | |
|-----|-----------------------|---|
| | | <pre>"user_name": "Alice", "app_name": "samples_test_client", "user_email": "abc@ascertia.com", "user_mobile": "00448004220562" }</pre> |
| 400 | Bad Request | |
| 500 | Internal Server Error | |

Table 11 – Users Profile

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|--|
| user_id | MANDATORY | String | User ID that identifies the user. |
| user_name | MANDATORY | String | It's the user name. |
| App_name | CONDITIONAL | String | Application name the user belongs to. |
| user_email | MANDATORY | String | User email |
| user_mobile | MANDATORY | String | User mobile number. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | A string with the description of the error_code. |

4.12 Get Device Registration Settings

This interface is used to get the device related settings configured in the RAS Profile for the devices of a user. This API should be invoked before registering a device to check which key length & key type will be used to generate an authorisation key-pair and other settings e.g. where to generate this key-pair i.e. in Device Secure Enclave or Software KeyStore/KeyChain.

| | | |
|---|-----------------------|----------------------|
| http://server:8778/adss/service/ras/v1/users/profile/device/settings | | |
| HTTP Verb | GET | |
| Accept | application/json | |
| Authorization | Bearer {access_token} | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |

| | | |
|-----|-----------------------|---|
| 200 | OK | <pre>{ "device_key_type": "ECDSA", "device_key_size": 256, "secure_element_required": true, "biometric_required": true, "allowed_devices": 10, "clock_tolerance_on_auth_cert": 10 }</pre> |
| 400 | Bad Request | |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 12 - Get Device Registration Settings

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------------|-------------|---------|---|
| device_key_type | MANDATORY | String | Key pair type to be generated in mobile device e.g. RSA Possible values are RSA & ECDSA. |
| device_key_size | MANDATORY | Integer | Key pair size e.g. 2048 |
| secure_element_required | MANDATORY | String | If set "TRUE" then the authorisation key pair must be generated inside device secure enclave otherwise software key store or KeyChain be used for key pair generation. If this flag is set to TRUE and the device does not support a Secure Element, then an error should be generated by mobile app. |
| biometric_required | MANDATORY | String | If set TRUE then device must have biometric (fingerprint, TouchID, FaceID etc.) support available on it otherwise Device PIN/Passcode be used to protect the generated keys. If this flag is set to TRUE and the device doesn't support biometric functionality, then an error should be generated by mobile app. |
| allowed_devices | MANDATORY | Integer | It defines the limit of devices to be registered. Default value is 0 for unlimited devices. |
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

4.13 Generate QR Code

This API will be used by the business application to generate a QR Code using the RAS Service. The RAS Service will generate a QR Code image for a user and send in response. The business application can display this QR code where user can scan it on his/her mobile device for authentication.

| | | |
|---|-----------------------|--|
| <a href="https://<server>:8778/adss/service/ras/v1/authentication/qrcode/{clientId}/{userId}?{format=png}&{size=264}">https://<server>:8778/adss/service/ras/v1/authentication/qrcode/{clientId}/{userId}?{format=png}&{size=264} | | |
| HTTP Verb | GET | |
| Content-Type | | |
| Accept | application/json | |
| Request Body | | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "format": "png", "size": "264", "qr_code": "<base64 encoded image>" }</pre> |
| 400 | Bad Request | |
| 500 | Internal Server Error | |

Table 13 – Generate QR Code

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|---------|---|
| {clientId} | MANDATORY | String | Client ID that is registered in the ADSS Client Manager. |
| {userId} | MANDATORY | String | User ID for whom the QR Code will be generated. |
| size | OPTIONAL | Integer | Size of the QR Code image in pixels. Since QR Code is in a square shape the parameter "size" will be used for both width and height of the image. Default size is 264 pixels Note: When invalid size is provided, 264 is used as default size. |
| format | OPTIONAL | String | Format of the QR Code image e.g. png/jpeg/bmp etc. Default format will be "png". The following formats are supported: <ul style="list-style-type: none"> png |

| | | | |
|--|--|--|---|
| | | | <ul style="list-style-type: none"> • jpg • bmp • jpeg • wbmp • gif <p>Note: When invalid format is provided, "png" is used as default format.</p> |
|--|--|--|---|

Response Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|----------------|---|
| format | MANDATORY | <i>String</i> | Format of the QR Code image e.g. png/bmp/jpg etc. |
| size | MANDATORY | <i>Integer</i> | Size of the QR Code image. |
| qr_code | MANDATORY | <i>String</i> | Base64 encoded image of the QR Code. |

4.14 Verify QR Code

This API will be used to verify a QR Code by RAS Service if user set the authentication mechanism QR code in RAS profile. Mobile app can use the QR code reader to scan the QR code. If QR code is verified successfully, the RAS Service will return the access and refresh tokens in response.

| <a href="https://<server>:8778/adss/service/ras/v1/authentication/qrcode">https://<server>:8778/adss/service/ras/v1/authentication/qrcode | | |
|---|--|---|
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {application_access_token} | |
| Request Body | <pre>{ "user_id":"jhon123", "qr_code":"Information extracted from QR code" }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "access_token":"2YotnFZFEjr1zCsicMWpAA", "refresh_token":"TRVfHTHcedfJGJFLGKKJ", "expires_in":3600 }</pre> |
| 401 | Unauthorized | If QR code is invalid or expired |
| 400 | Bad Request | |

| | | |
|-----|-----------------------|--|
| 500 | Internal Server Error | |
|-----|-----------------------|--|

Table 14 – Verify QR Code

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|--|
| user_id | MANDATORY | String | User ID that identifies the user. |
| qr_code | MANDATORY | String | The QR code that needs to be verified. |

Response Parameters

| Parameters | Presence | Value | Description |
|---------------|-----------|--------|---|
| access_token | MANDATORY | String | Access Token generated after verifying the QR code. |
| refresh_token | MANDATORY | String | Refresh token will be used to get the new access token without sending any credentials. |
| expires_in | MANDATORY | String | It's token expiry mentioned in the seconds. |

4.15 Register Device for Push Notification

This API is used to register the mobile device for push notification by RAS Service. It takes the device token from the mobile application and stores in ADSS RAS to send the push notification while generating the authorization request.

| | | |
|---|--|---------------------------------------|
| <a href="https://<server>:8778/adss/service/ras/v1/authorization/push/notification">https://<server>:8778/adss/service/ras/v1/authorization/push/notification | | |
| HTTP Verb | POST | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {user_access_token} | |
| Request Body | <pre>{ "device_token": "2YotnFZFEjr1zCsicMWpAA ", "os_type": "ANDROID/IOS" }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |
| 401 | Unauthorized | Invalid or expired user access token. |
| 400 | Bad Request | Device token is missing. |

| | | |
|-----|-----------------------|--|
| 500 | Internal Server Error | |
|-----|-----------------------|--|

Table 15 – Register Device for Push Notification

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|---|
| device_token | MANDATORY | String | It's the device token which needs to be sent on server to register the mobile device for push notification. |
| os_type | MANDATORY | String | OS type can be Android or iOS. |

4.16 Delete Device for Push Notification

This API is used to delete the registered mobile device for push notification by RAS Service.

| | | |
|---|---|----------------------|
| <a href="https://<server>:8778/adss/service/ras/v1/authorization/push/notification/{device_token}">https://<server>:8778/adss/service/ras/v1/authorization/push/notification/{device_token} | | |
| HTTP Verb | DELETE | |
| Content-Type | application/json | |
| Accept | application/json | |
| Authorization | Bearer {device_token} | |
| Request Body | <pre>{ "device_token": "2YotnFZFEjr1zCsicMWpAA" }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | |

Table 16 – Delete Device for Push Notification

Request Parameters

| Parameters | Presence | Value | Description |
|--------------|-----------|--------|---|
| device_token | MANDATORY | String | It's the device token which needs to be sent on server to register the mobile device for push notification. |

Response Parameters

| Parameters | Presence | Value | Description |
|-------------------|-------------|--------|----------------------------|
| error_code | CONDITIONAL | String | The error code. |
| error_description | CONDITIONAL | String | Error description message. |

5 Signature Activation Data (SAD) – Body Structure

The body structure of SAD XML is explained in the table below:

| | |
|----------------|--|
| Body Structure | <pre> <AuthorisationData> <OriginatorID>Virtual_CSP_Client</OriginatorID> <UserID>olcayatli@gmail.com</UserID> <CertificateID>416edc72-6c63-45aa- bb34a373102234df</CertificateID> <TransactionID>980551837300673581</TransactionID> <Salt>924552495291565632</Salt> <MetaData> <DisplayText>Data to be displayed</DisplayText> <DeviceID>ad1e60a6-5e23-4b52-a127-27f41c224c05</DeviceID> <!-- Only required if Go>Sign Mobile App is being used for authorisation --> <!-- AuthenticationData is required if external IdP is being user for authentication --> <AuthenticationData type="jwt"> <!-- type can be "jwt" or "saml" - -> eyJ0eyJ0eyJ0NiJ9.eyJ1ceyJ1ceyJ1cI6IjIifX0.DeWt4QuDeWt4QuDeWt4QuZXs o</AuthenticationData> </MetaData> <NumSignatures>100</NumSignatures> <Documents> <Document id="b81e040a-a4d8-4134-92ff-2d4bf5e9116d"> <Name>b81e040a-a4d8-4134-92ff-2d4bf5e9116d</Name> <DigestValue>ypkP9L2tZO2JdfNr4X4X5SRur529uJqykdc5q5HDSiLNiYcLrys00 S/H31yb8QZS&#xD;SOBYsF1VSj9/SKUqrhsUC5oEc/gr</DigestValue> </Document> </Documents> <ValidityPeriod> <ValidFrom>2019-12-07T18:25:37</ValidFrom> <ValidTo>2019-12-07T18:42:17</ValidTo> </ValidityPeriod> <Signature> <DigestMethod>SHA256</DigestMethod> <SignatureValue>MEUCID/kiJWAIqzwOp/hi+FUbJwsjdcsEoBNwlIF8sXA8XbDAi EAkGgQomyoJ2iR0ra9KGBFW/zXi6tbsn5M49YiaPNc+L8=</SignatureValue> </AuthorisationData> </pre> |
|----------------|--|

6 Get Profile Information

This interface returns the information of a RAS profile e.g. all settings configured in that profile. The business application will send the profile ID and client ID in request and RAS will return the information of that profile in response.

Exposed for: Business Applications

| https://server:8779/adss/service/ras/v1/profile/info | | |
|---|--|--|
| HTTP Verb | POST | |
| Accept | application/json | |
| Request Body | <pre>{ "profile_id": "adss:ras:profile:001 ", "client_id": "samples_test_client" }</pre> | |
| | | |
| Status Code | Message | Response Body |
| 200 | OK | <pre>{ "profile_id": "adss:ras:profile:001", "profile_name": "adss:ras:profile:001", "profile_status": "ACTIVE", "basic_authentication": true, "oauth2_authentication": true, "credentials_authorisation_method": "IMPLICIT", "authentication_with_qr_code": false, "no_authentication": false, "sam_profile": { "profile_id": "adss:sam:profile:001", "profile_name": "adss:sam:profile:001", "profile_status": "ACTIVE", "crypto_profile": "utimaco", "key_type": "RSA", "key_size": 2048, "kak_size": 2048, "signature_padding_scheme": "PKCS1", "compute_hash_at_signing": true, "hash_algorithm": "SHA256", "bulk_signing_allowed": false, "number_of_hashes": 0, "device_key_type": "ECDSA", "device_key_size": 256, "secure_element_required": true, "biometric_required": true, "allowed_device": 10, "user_authorisation_settings": "ONCE", "sole_control_type": "2", "authorisation_request_expiry": "100", "authorisation_request_expiry_unit": "SECONDS", "clock_tolerance_on_auth_cert": "10" }, "saml_assertion": { "idp_signing_certificate": "", "identify_user_id": "SAML_ATTRIBUTE_NAME", </pre> |

| | | |
|-----|-----------------------|---|
| | | <pre> "identify_user_attribute": "abc" }, "authentication_with_otp": { "sms_otp": true, "email_otp": true }, "authorisation_certificate": "auth_cert_alias", "delegated_authentication": true, "external_idp": "IdP Connector" } </pre> |
| 400 | Bad Request | Device token is missing |
| 403 | Forbidden | |
| 404 | Not Found | |
| 500 | Internal Server Error | |

Table 17 – Get Profile Information

Request Parameters

| Parameters | Presence | Value | Description |
|------------|-----------|--------|---|
| client_id | MANDATORY | String | Client's Originator ID which is configured in ADSS Console > Client Manager. Client ID is required because only legitimate clients can get a profile's information. |
| profile_id | MANDATORY | String | Profile ID whose information is required in response. |

7 Error Code List

Below table contains the error codes for RAS business and mobile interfaces.

| Errors | |
|--------|---|
| Error | Error Description |
| 58001 | An internal server error occurred while processing the request - see the RAS service debug logs for details |
| 58002 | Service is not available - Try later |
| 58003 | Failed to process request - RAS service is not enabled in license |
| 58004 | Failed to process request - RAS service license has expired |
| 58005 | Failed to process request - RAS service is not enabled in system |
| 58006 | Failed to process request - RAS service is not allowed |
| 58007 | Failed to process request - Client ID does not exist |
| 58008 | Failed to process request - User ID does not exist |
| 58009 | Failed to process request - User ID already exists |
| 58010 | Failed to process request - Key alias does not exist |
| 58011 | Failed to process request - Transaction ID does not exist |
| 58012 | Failed to process request - Client ID not found in the request |
| 58013 | Failed to process request - User ID not found in the request |
| 58014 | Failed to process request - Key alias not found in the request |
| 58015 | Failed to process request - Subject DN not found in the request |
| 58016 | Failed to process request - User password not found in the request |
| 58017 | Failed to process request - Key length not found in the request |
| 58018 | Failed to process request - Key algorithm not found in the request |
| 58019 | Failed to process request - User name not found in the request |
| 58020 | Failed to process request - User password not found in the request |
| 58021 | Failed to process request - User mobile number not found in the request |
| 58022 | Failed to process request - Key alias exceeds the allowed limit |
| 58023 | Failed to process request - User ID exceeds the allowed limit |
| 58024 | Failed to process request - User name exceeds the allowed limit |
| 58025 | Failed to process request - User password exceeds the allowed limit |
| 58026 | Failed to process request - Invalid user mobile number |
| 58027 | Failed to process request - Invalid user email |
| 58028 | Failed to process request - Invalid user status |
| 58029 | Failed to process request - RAS profile does not exist or marked inactive |
| 58030 | Failed to process request - User certificate not found in the request |

| | |
|-------|---|
| 58031 | Failed to process request - Profile ID not found in the request |
| 58032 | Failed to process request - Invalid client ID |
| 58033 | Failed to process request - User's new password not found in the request |
| 58034 | Failed to process request - SMS OTP not found in the request |
| 58035 | Failed to process request - Email OTP not found in the request |
| 58036 | Invalid string parameter - refresh_token |
| 58037 | Failed to process request - Invalid refresh token |
| 58038 | Failed to process request – Invalid/expired access token |
| 58039 | The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed. |
| 58040 | Failed to process request - Basic authentication is not enabled in RAS profile |
| 58041 | Failed to process request - SAML authentication is not enabled in RAS profile |
| 58042 | Failed to process request - Missing (or invalid type) string parameter token |
| 58043 | Failed to process request - Invalid string parameter token_type_hint |
| 58044 | Failed to process request - Invalid string parameter token |
| 58045 | Failed to process request - Client ID is not configured for certification service settings in RAS service manager |
| 58046 | Failed to process request - Certification profile is not configured for certification service settings in RAS service manager |
| 58047 | Failed to process request - Certification service address is not configured for certification service settings in RAS service manager |
| 58048 | Failed to process request - Unable to get device certificate from certification service |
| 58049 | Failed to generate access token - HMAC Key not configured in RAS service manager |
| 58050 | Failed to process request - User Email not found in the request |
| 58051 | Failed to process request - Client ID is not configured for default settings in RAS service manager |
| 58052 | Failed to process request - Device ID not found in the request |
| 58053 | Failed to process request - Push notification token not found in the request |
| 58054 | Failed to process request - OS type not found in the request |
| 58055 | Missing (or invalid type) string parameter credentialID |
| 58056 | Missing (or invalid type) integer parameter numSignatures |
| 58057 | Invalid parameter numSignatures |
| 58058 | Invalid request parameter - numSignatures doesn't match with no of hashes in hash array |
| 58059 | Invalid request parameter - no of documents in clientData doesn't match with no of hashes in hash array |

| | |
|-------|--|
| 58060 | Missing parameter hash |
| 58061 | Failed to authorise user credentials - Request timeout for mobile authorisation |
| 58062 | Failed to authorise user credentials - User cancelled mobile authorisation |
| 58063 | Invalid parameter credentialID |
| 58064 | Missing (or invalid type) string parameter SAD |
| 58065 | Invalid parameter SAD |
| 58066 | Empty hash array |
| 58067 | Invalid Base64 hash string parameter |
| 58068 | Missing (or invalid type) string parameter signAlgo |
| 58069 | Invalid parameter signAlgo |
| 58070 | Missing (or invalid type) string parameter hashAlgo |
| 58071 | Invalid parameter hashAlgo |
| 58072 | Failed to validate SAML assertion - Invalid base64 data |
| 58073 | Failed to validate SAML assertion - Not comply with SAML 2.0 schema |
| 58074 | Failed to validate SAML assertion - Unable to parse SAML assertion |
| 58075 | Failed to validate SAML assertion - Validity period expired or not yet valid |
| 58076 | Failed to validate SAML assertion - Server certificate does not match with the certificate configured in RAS Profile |
| 58077 | Failed to validate SAML assertion - Multiple or no attributeValue found |
| 58078 | Failed to validate SAML assertion - Invalid Signature |
| 58079 | Failed to process request - User status is blocked or inactive |
| 58080 | Failed to process request - Certificate chain not found in the request |
| 58081 | Failed to process request - Invalid certificate chain |
| 58082 | Failed to process request - Client secret not found in the request |
| 58083 | Failed to authorise user credentials - An internal server error occurred during signature computation |
| 58084 | Failed to process request - Device CSR not found in the request |
| 58085 | Failed to process request - Invalid device CSR |
| 58086 | Failed to process request - Device information not found in the request |
| 58087 | Failed to process request - Device ID not found in the request |
| 58088 | Failed to process request - Device name not found in the request |
| 58089 | Failed to process request - SAD not found in the request |
| 58090 | Failed to process request - Request ID not found in the request |
| 58091 | Failed to process request - Invalid request |

| | |
|-------|---|
| 58092 | Failed to process request - Either request ID is invalid or the transaction is expired |
| 58093 | An internal server error occurred - please contact your service provider |
| 58094 | Failed to process request - User mobile exceeds the allowed limit |
| 58095 | Failed to process request - User email exceeds the allowed limit |
| 58096 | Failed to process request - Configurations for SMS/Email OTP(s) not available |
| 58097 | Failed to process request - No OTP(s) found in request |
| 58098 | Failed to process request - QR Code authentication is not allowed for this RAS profile |
| 58099 | Failed to process request - QR Code is invalid or expired |
| 58100 | Failed to process request - Missing parameter Grant type |
| 58101 | Failed to process request - Invalid parameter Grant type |
| 58102 | Failed to process request - Missing parameter authorization code |
| 58103 | Failed to process request - Invalid parameter authorization code |
| 58104 | Failed to process request - Missing parameter Redirect URI |
| 58105 | Failed to process request - Missing Redirect URI |
| 58106 | Failed to process request - Authorization code is invalid or expired |
| 58107 | Failed to process request - Scope is missing or invalid |
| 58108 | Failed to process request - Response type is missing or invalid |
| 58109 | Missing (or invalid type) integer parameter certificates |
| 58110 | Information message is not present for the credential authorization page |
| 58111 | Failed to validate signature activation data - XML schema validation failed |
| 58112 | Failed to validate signature activation data - Failed to parse the XML |
| 58113 | Failed to process request - Missing parameter 'PIN' |
| 58114 | Failed to process request - Missing or invalid parameter 'OTP' |
| 58115 | Failed to process request - Missing or invalid parameters 'PIN' and 'OTP' |
| 58116 | Failed to process request - Invalid PIN |
| 58117 | Failed to process request - OTP is invalid or expired |
| 58118 | Failed to Sign SAD - Authorisation Certificate is not configured in RAS profile |
| 58119 | Failed to process request - Explicit credentials authorisation with OTP must be selected in profile |
| 58120 | Failed to process request - Extend transaction is not allowed. Profile is set to require user authorisation for every transaction |
| 58121 | Failed to process request - The limit of devices you can register has been reached |

| | |
|----------------|--|
| 86000 | Failed to authenticate client - TLS client authentication certificate has expired |
| 86001 | Failed to authenticate client - TLS certificate CN does not match with Client ID |
| 86002 | Failed to authenticate client - TLS client certificate is revoked |
| 86003 | Failed to authenticate client - revocation status for TLS client certificate is unknown |
| 86004 | Failed to authenticate client - Client ID does not match with the client identified by TLS client certificate |
| 86005 | Failed to authenticate client - TLS client certificate does not match with the configured client certificate |
| 86006 | Failed to authenticate client - request signing certificate has expired |
| 86007 | Failed to authenticate client - request signing certificate is revoked |
| 86008 | Failed to authenticate client - revocation status for request signing certificate is unknown |
| 86009 | Failed to authenticate client - request signing certificate does not match with the configured client certificate |
| 86010 | Failed to authenticate client - Client ID does not match with the client identified by the request signing certificate |
| 86011 | Failed to authenticate client - Client ID does not exist |
| 86012 | An error occurred while communicating with database - see the service debug logs for details |
| 86013 | An error occurred when checking the certificate revocation status - see the service debug logs for details |
| 86014 | An internal error occurred while authenticating the client - see the service debug logs for details |
| 86015 | Failed to authenticate client - Client ID is not found in the request |
| 86016 | Failed to process request - Request signing certificate is not trusted |
| 86017 | Failed to authenticate client - client is marked inactive |
| 86018 | Failed to authorise client - service is not allowed to this client |
| 86019 | Failed to authorise client - service profile does not exist |
| 86020 | Failed to authorise client - service profile is inactive |
| 86021 | Failed to authorise client - profile is not allowed to this client |
| 86022 | Failed to authorise client - default profile not configured and neither found in request |
| 86023 | Failed to authorise client - default profile is inactive |
| 86024 | Failed to authorise client - client secret is invalid |
| internal_error | An internal server error occurred while processing the request |
| invalid_csr | CSR is invalid |

| | |
|-----------------------|--|
| invalid_otp | OTP is either invalid or expired |
| missing_csr | CSR is missing in the request |
| missing_device_id | Device ID is missing in the request |
| missing_device_info | Device information is missing in the request |
| missing_device_name | Device name is missing in the request |
| missing_request_id | Request ID is missing in the request |
| refresh_token_revoked | Refresh token is either invalid or expired |

Table 16 - Error Codes

8 Use Cases of RAS Service

There are multiple use cases that can be configured in ADSS RAS Service. List of such use cases are explained below:

8.1 Restrict Access to RAS API's

There are two major types of API's exposed by RAS Service for registration and signing purposes, these include:

- **Mobile Application API's:** These APIs are used for the registration of user mobile device for remote authorisation as well as allow the user to receive and authorise the remote signing requests. These API's are used by Mobile Applications e.g. Go>Sign Mobile Application.
- **Business Applications API's:** These APIs are used for user registrations and signing operations. These API's are used by Business Applications.

Since these APIs are used by different types of clients (Mobile Apps, Business Apps), there can be scenarios where you want to restrict a mobile application to access the Business APIs and similarly a business application to access the Mobile APIs of the RAS Service. This can be achieved by deploying RAS Gateways in front of the RAS Service and configuring these gateways to allow access to a certain type of APIs.

Configure RAS Gateway to allow access to Mobile APIs only:

To restrict Go>Sign mobile application to access only the Mobile APIs, a RAS Gateway will be deployed in front of the RAS Service and Go>Sign mobile application will be integrated with the RAS Gateway instead of the RAS Service. In this deployment scheme, the Go>Sign Mobile Application will communicate with RAS Gateway and RAS Gateway will forward requests to the backend RAS service. In order to configure RAS Gateway to accept requests for Mobile API's only, follow the instructions below:

1. On RAS Gateway, go to location **<ADSS-Installation-Dir>/service/server/webapps/service/WEB-INF**
2. Open the "web.xml" file
3. Search the RAS Service package using the string "**com.ascertia.adss.service.ras**"

```

<servlet>
  <servlet-name>Jersey REST Service</servlet-name>
  <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
  <init-param>
    <param-name>jersey.config.server.provider.packages</param-name>
    <param-value>
      com.ascertia.adss.service.ras, com.ascertia.adss.service.sam,
      com.ascertia.adss.service.csp, com.ascertia.adss.service.hmac
    </param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>

```

4. Replaces the above RAS package with the following packages:
 - a. com.ascertia.adss.service.ras.mobile
 - b. com.ascertia.adss.service.ras.qrcode
 - c. com.ascertia.adss.service.ras.status

```

<servlet>
  <servlet-name>Jersey REST Service</servlet-name>
  <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
  <init-param>
    <param-name>jersey.config.server.provider.packages</param-name>
    <param-value>
      com.ascertia.adss.service.ras.mobile, com.ascertia.adss.service.ras.qrcode,
      com.ascertia.adss.service.ras.status, com.ascertia.adss.service.sam,
      com.ascertia.adss.service.csp, com.ascertia.adss.service.hmac
    </param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>

```

5. Save the updated 'web.xml' file and close it.
6. Restart the ADSS Services.

Configure RAS Gateway to allow access to Business API's only:

Similarly, we can also configure RAS Gateway to allow access to only Business APIs. This Gateway will be deployed in front of RAS Service and Business Applications will interact with this gateway. In this deployment scheme, the Business Application will communicate with RAS Gateway and RAS Gateway will forward requests to the backend RAS service. In order to configure RAS Gateway to accept requests for Business API's only, follow the instructions below:

1. On RAS Gateway, go to location **<ADSS-Installation-Dir>/service/server/webapps/service/WEB-INF**
2. Open the "web.xml" file
3. Search the RAS Service package using the string "**com.ascertia.adss.service.ras**"

```

<servlet>
  <servlet-name>Jersey REST Service</servlet-name>
  <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
  <init-param>
    <param-name>jersey.config.server.provider.packages</param-name>
    <param-value>
      com.ascertia.adss.service.ras, com.ascertia.adss.service.sam,
      com.ascertia.adss.service.csp, com.ascertia.adss.service.hmac
    </param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>

```

4. Replaces the above RAS package with the following packages:
 - a. com.ascertia.adss.service.ras.login
 - b. com.ascertia.adss.service.ras.csc
 - c. com.ascertia.adss.service.ras.keypair
 - d. com.ascertia.adss.service.ras.qrcode
 - e. com.ascertia.adss.service.ras.users
 - f. com.ascertia.adss.service.ras.status

```
<servlet>
  <servlet-name>Jersey REST Service</servlet-name>
  <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
  <init-param>
    <param-name>jersey.config.server.provider.packages</param-name>
    <param-value>
      com.ascertia.adss.service.ras.login, com.ascertia.adss.service.ras.csc,
      com.ascertia.adss.service.ras.keypair, com.ascertia.adss.service.ras.grcode,
      com.ascertia.adss.service.ras.users, com.ascertia.adss.service.ras.status
      com.ascertia.adss.service.sam, com.ascertia.adss.service.csp,
      com.ascertia.adss.service.hmac
    </param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
```

5. Save the updated 'web.xml' file and close it.
6. Restart the ADSS Services.

*** End of Document ***