Web Services Authentication API Specification

RemitONE Money Transfer Management Solution

Version 1.23
August 2018

Table of Contents

Introduction

The RemitONE WebServices Infrastructure uses the REST protocol. Our services use the POST method for invoking the service, and hence all data must be provided as POST variables. The response is provided as structured XML.

This document outlines the input and output parameters for each of the RemitONE WebServices.

Connection Parameters

The BASE_URL is the URL at which the WebServices are hosted. This BASE_URL is then followed by the required GROUP name, and then the required METHOD name.

For example,
BASE_URL = https://www.remitone.com/ws/

GROUP = auth
METHOD = getSeed

This would give a complete URL of :
https://www.remitone.com/ws/auth/getSeed

The client application would need to make a POST to this URL to invoke this particular WebService. *In the specification below, a * indicates a required input parameter.*

Every invocation must contain the following parameters (as POST variables) :

Name	Type	Notes
username *	Text	
password *	Text	
pin *	Text	

These will be provided to you when your WebService Account is activated.

All other parameters will depend on the particular WebService being invoked.

Response Format

A Successful Response will always contain the following XML structure :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        [Specific response for the particular Web Service]
    </result>
</response>
```

An Error Response is as follows :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>FAIL</status>
    <result>
        <message>[Error message]</message>
    </result>
</response>
```

An Error Response with Validation errors is as follows :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>FAIL</status>
    <result>
        <message>VALIDATION FAILED</message>
        <errors>
            <error>[Validation error message]</error>
            <error>[Validation error message]</error>
            <error>[Validation error message]</error>
        </errors>
    </result>
</response>
```

Web Service Method Details

Authentication API

getSeed

Group : **auth**
Method : **getSeed**
This method provides a random seed which can be used to login an Agent from an external site to the internal ARM system.

Input fields:

Name	Type	Notes
agent_username *	Text	The user name of the Agent who wants to login to the ARM.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <seed>1234567890123</seed>
    </result>
</response>
```

---

getAuthToken

Group : **auth**
Method : **getAuthToken**
This method provides an encrypted token which can be used to login into the ARM system without a password.

Input fields:

Name	Type	Notes
encrypted_string *	Text	The Encrypted String, which contains the seed, agent username and a pre shared key.

Example Output XML :

```
<?xml version="1.0" encoding="utf-8"?>
<response>
    <status>SUCCESS</status>
    <result>
        <token>12asd345asd6789012asd3</token>
    </result>
</response>
```

Testing Web Services Functionality

It is possible to test the output of the WebServices if the system is in debugging mode. The URL for accessing forms for each of the WebServices is :

SYSTEM_URL/wstest/index.html

For example, https://www.remitone.com/wstest/index.html

You can enter values for each of the WebService Forms and the result will be presented back as neatly formatted HTML.

If the system is in live mode, then the result will be returned as pure XML, and hence you can still use the 'View Source' option in the Web Browser to see the XML that is returned.

How to use the Authentication API

The authentication API can be used to provide Single Sign On functionality for Agents on the ARM, where your agents login to another system and then access the ARM without the need to login again for a second time on the R1 ARM.

The process from the user's perspective would be as follows:

1. The agent logs into the portal or other system used by the company. The password can be different to the one used on the R1 system.
2. The portal provides a link to the R1 ARM so that the user can move to the R1 system
3. On doing so, the ARM will open with the user logged in already, so there is no need to login twice.

The technical process is as follows:

1. The user logs into the company portal, and thus completes authentication there.

2. The portal will call the R1 Authentication WS method getSeed() and provide the agent's username as a POST parameter (agent_username)

3. The result of the getSeed call is then used to obtain a token by calling getAuthToken. This uses the seed from step 2 along with the agent's username and a pre-shared key.

4. The token from step 3 is then used to generate a URL for the user to logon to the ARM. This URL will be the standard system URL with a GET parameter added which is:
token=<token_from_step_3>
So for example:
https://system.company.com/?token=XXXXXXXXXXX
On accessing this URL the agent will be automatically logged in.