Afteroffice Universal Authentication API 1.0b (internal)

Afteroffice Universal authentication API provides method for clients (none-browser applications) to authenticate with AVO framework and services thru unified, centralized method, covering all servers at AVO network and services offered to third party.

The authentication server:

==========

`auth.agnx.com`

The authentication URL:

==========

`/-/vo/public/login.authenticate`

The protocol:

==========

http or https (self-signed)

The HTTP method:

==========

The HTTP query supports for both GET and POST method. Samples in this document use GET for easy illustration, not endorsement. It is advisable to use POST method.

Keys supported:

==========

`name, pswd, service, format, keys`

Sample:

==========

`http://auth.agnx.com/-/vo/public/login.authenticate?name=user@afteroffice.com&pswd=x1a2s3d`

Keys description:

==========

name: the login name, usually the full email. If email is not provided, the "service" should be given (for TM eStorage authentication).

pswd: the login password, in plain.

service: [optional] use in TM eStorage authentication only, where user have to provide their service type: streamyx, 1515, HSIA, business, webhosting, myapp, tmconsumer.

format: [option] returning format, default is "lines". Supports for "XML" and "JSON".

keys: [optional] additional keys to obtain from the user profile, separated by comma. Eg: ucargroup,

Returning data:

==========

Server returns data in *plain text*, *XML* or *JSON* format (controlled by the optional key *format* in the query URL). The line separator is following UNIX convention CR (ASCII 10).

Plain text format is as follow:

```
key1<colon>data 1<cr>
key2<colon>data 2<cr>
....
key[n]<colon>data n<cr>
```

Note that CR in the data will be encoded as "\n".

status: the authentication status, "ok" or "error". More detail in "description".

description: the login status description (english) when the login has failed.

session: the session ID to use for queries over AVO modules.

server: the server to send the additional queries to - the user's server, where all user's data is.

email: the user email

fullname: the user full name

iolimit: the email storage limit (in bytes)

storagelimit: the disk limit in storage (in bytes)

cookie: the token submit back to the server on HTTP query as Cookie. When cookie is available at the return data, the application must use it to perform HTTP query to the server.

Login flow:

==========

Client send the following URL to the server:

```
http://auth.agnx.com/-/vo/public/login.authenticate?
name=user@afteroffice.com&pswd=x1a2s3d
```

If the authentication successful, server responds:

```
status:ok
description:Login OK
fullname:ABU
session:SIDxue88391XDES
storagelimit:10240
iolimit:5120
email:user@afteroffice.com
```

If the password is invalid, server responds:

```
status:error
description:Invalid password
```

Possible error descriptions:

```
Require name or password
Invalid host
Invalid name
Invalid password
Host disabled
Login disabled
Login expired
```

By

2012-04 Afteroffice.com