REST API - Authentication

**Overview**

The Higher Logic API has been designed for ease-of-communication by both Client-side JavaScript and Server-side (PHP, C#, Perl, etc.) technologies. The specific technologies utilized by any given organization will vary; therefore the API has been created to enable access via HTTP for any development platform.

<https://api.connectedcommunity.org/Help/Api/GET-api-v2.0-Contacts-GetWhoAmI>

https://network.futureearth.org/test/laurel

**The Identity and Access Management (IAM) System**

The Higher Logic Identity and Access Management (**IAM**) System is built around Context- and Organization-Specific access to data within the Connected Community Platform. The IAM System is comprised of a *multi-usage* IAMKey (Guid) that must be passed to the API as a Request Header in one of the combinations described below. This Key is *multi-usage* in that it can be used to either identify a specific user within an Organization’s Community Site, or simply to identify the Organization where the API request is being directed. At this time, you organization’s IAMKey(s) will be provided to you upon request – as it must be configured by the Higher Logic team.

Here are the 3 IAMKey use cases:

1. **IAMKey + Password**:

When combined with a Password, an IAMKey resolves to a specific Contact within a specific Connected Community Site.

**Conceptually**:

|  |  |
| --- | --- |
| Headers | Resolves To |
| HLIAMKey + HLPassword | ‘Client X’: ‘User Y’ : Access Rights for ‘User Y’ |

**Sample Request**

GET https://api.connectedcommunity.org/api/v2.0/Contacts/GetWhoAmI HTTP/1.1

Host: api.connectedcommunity.org

Accept: application/json

Accept-Encoding: gzip,deflate,sdch

Accept-Language: en-US,en;q=0.8

Content-Type: application/json

**HLIAMKey**: BE5F82F0-B463-41AB-A349-04E976A8D552

**HLPassword**: <PRE-ESTABLISHED\_PASSWORD>

1. **IAMKey + Authentication Token**:

A Higher Logic Authentication Token contains encrypted information providing a “lease” on data access rights specific to the Contact for whom the Token was originally created. In this use case, the IAMKey resolves to a specific Community Site, which must match the Community Site associated with the Contact encrypted within the Authentication Token. The process to obtain an Authentication Token is described in the section [below](#LoginEndpoint).

**Conceptually**:

|  |  |
| --- | --- |
| Header | Resolves To |
| HLAuthToken | ‘Client X’ : Access Rights for ‘User Y’ |
| HLIAMKey | ‘Client X’ |

**Sample Request**

GET https://api.connectedcommunity.org/api/v2.0/Contacts/GetWhoAmI HTTP/1.1

Host: api.connectedcommunity.org

Accept: application/json

**HLIAMKey**: 5A48120A-A57E-45Y2-86N5-34DB86B570A9

**HLAuthToken**: ofMS0rORUoGdJrY7IENdaOtpPSqxex9+q8ao+NpVIpB4lojXgoVioANWwW2KGuXMnHwecqDnPugq+JY2pYXel5NyNBDc+7oJUQ4u8z3vfLOkljK/jnj9ps0eu90Ls2Oi

1. **IAMKey + *Authentication Browser Cookie***:

The *Authentication Browser Cookie* contains the same information as the Authentication Token but it is not explicitly sent or generated by the consumer of the API. This Cookie is only set in the end-user’s Browser when they successfully log into the Connected Community site. Once this Cookie is set, it contains an Authentication Token with access information specific to the User who has most recently logged into the Community Site.

If multiple users have logged into one or more Connected Community Sites from the same Browser, the Browser will store the latest Authentication Cookie for each Community Site the user has logged into. See [below](#SuccessfulLogin) for additional information on establishing an API Browser Cookie.

The Browser Cookie Authentication method is in place specifically to allow an easy way for clients to utilize the API via Client Script. Here is an example JavaScript AJAX call to the API:

$.ajax({

                type: 'GET',

                url: 'https://api.connectedcommunity.org/api/v2.0/Contacts/GetWhoAmI',

                datatype: "application/json",

                headers: {

                    "HLIAMKey": "5A48120A-A57E-45Y2-86N5-34DB86B570A9"

                },

                xhrFields: {

                    withCredentials: true

                },

                success: function (result) {

                     alert("Authenticated as: " + result.DisplayName);

                },

                error: function (error) {

                    alert('Call failed.');

                }

});

**Authentication and Establishing the Authentication Token**

There are 2 Methods by which a user can establish an Authentication Token to be used in calls to the Higher Logic API:

1. Making a call to the Login Endpoint
2. Successful Login to a Connected Community Site

### **1. Login Endpoint**

To retrieve an Authentication Token (valid for 24 hours from the time of creation) explicitly, make a call to the Login Endpoint. An example *GET* Request, sent via Chrome:

GET https://api.connectedcommunity.org/api/v2.0/Authentication/GetLogin?**username=testuser@higherlogic.com&password=p@ssw0rd** HTTP/1.1

Host: api.connectedcommunity.org

Cache-Control: max-age=0

Accept: application/json

Accept-Encoding: gzip,deflate,sdch

**HLIAMKey: 01243024-DBAA-4E89-A704-9227859D59A4**

The Response for the above Request is:

{"Token":"qPQUKFywkDP0GwZhFBozBaxVzp2zzdiXpXq/U7tRZki5643mPMcW4k2U7ZeCvwFhTYxZgztLk1GCnosie8ZIevJrEYJLXBJb+o/f9BDb3U/SqmJGPbzrK5rk989rBQLX"}

This Token can then be used with the *Authentication Token* Mode described [above](#IAMKeyAndAuthenticationToken), where it would be passed to the API as the **HLAuthToken** Header.

As an alternate to the “GetLogin” endpoint, there is a POST Endpoint for Login, to allow the inclusion of the username/password in the Request Body instead of the Querystring. Here is an example of the usage:

POST https://api.connectedcommunity.org/api/v2.0/Authentication/Login HTTP/1.1

Host: api.connectedcommunity.org

Accept: application/json

Content-Type: application/json

**HLIAMKey: 01243024-DBAA-4E89-A704-9227859D59A4**

**{"username":"testuser@higherlogic.com","password":"p@ssw0rd"}**

The Response will be the same as for the GET Request:

{"Token":"qPQUKFywkDP0GwZhFBozBaxVzp2zzdiXpXq/U7tRZki5643mPMcW4k2U7ZeCvwFhTYxZgztLk1GCnosie8ZIevJrEYJLXBJb+o/f9BDb3U/SqmJGPbzrK5rk989rBQLX"}

### **2. Successful Login to a Connected Community Site**

In order to facilitate calls to the API via JavaScript, a mechanism is in place that will automatically generate an Authentication Token and set it as a Cookie under the Higher Logic API’s domain. This Cookie will be automatically read by the API when a request is serviced, and removes any need for the Client Script to request and explicitly submit an Authentication Token.

When Requests are sent to the API with a valid IAMKey (as an HLIAMKey Header), the API will automatically be able to determine what user is making the request based upon the last user to have logged into the Community Site associated with the passed IAMKey.

**API Endpoints and Usage**

The Higher Logic API is accessible from the base URL: <https://api.connectedcommunity.org>. A standardized Help section, describing all available operations and samples of the formatting of Requests and Response objects can be found here: <https://api.connectedcommunity.org/v2.0/Help>.

Remember, from the Authentication discussion above, there are numerous ways to be authenticated for any given request. So, it is possible to use any of the Authentication Methods for any of the operations described in the documentation. The only caveat to this is the “Login” Endpoint, which requires only the IAMKey and a Username + Password in either the Querystring or the POST Body, depending on which Login Endpoint is called.

The most current supported Operation Endpoints can be found at the Help link, above