How to put an OpenAthens sign-in on your portal

If you want to have users sign into OpenAthens on one of your own pages, such as a library portal, you can do so using our API. As you can imagine it is not practical to provide code samples for this however we can summarise the steps you need to perform in order, as well as linking to the relevant sections of the API documentation.

Prerequisites

- Some programming experience.
- Familiarity with the principals of an API.
- Familiarity with the portal or site where you will be putting the login.
- An API key - see: API keys.
- A secure form on your website to receive user credentials.

Process

1. Authenticate the user:
    a. Encode the username and password the end-user submitted in a standard HTTP authenticate header
    b. Send as a GET to https://login.openathens.net/api/v1/example.org/

       For details see API usage examples.
2. If you do not receive a `204 No Content` response: stop.

   From this point on you will use your own API key.
3. Identify the organisation of the user (if all your users are directly under your domain, this will be a constant but coding to look it up is better practice and allows greater flexibility later).
    a. Make a GET request to https://login.openathens.net/api/v1/example.org/account/query?username=USERNAME
    b. There will be an 'id' in the 'organisation' section of the response
    c. There will also be other information there that may be of use to you

       For details see Fetching individual accounts
4. Next you obtain a session initiator URL using the submitted username* by making a GET request to https://login.openathens.net/api/v1/example.org/organisation/**<id>**/account/session?username=USERNAME&returnUrl=https%3A%2F%2Fexample.org%2Fmy-app
    a. where <id> is the organisation id you obtained in the previous step
    b. and returnURL is an address in your application or site where the user should be sent next after establishing a session.

       For details see Generating authentication tokens for end-users via the API
5. The response will include a 'sessionInitiatorUrl'. This URL includes a time limited token.
6. Send the user to the sessionInitiatorUrl returned in the previous step via a 302 redirect.
    a. The user will then be signed in.

Anything to watch out for?

You must ensure that your application does not expose your API key to the end user.

The returnUrl must be to your own application or page rather than a resource because the status parameter will be unexpected anywhere else.

* As well as submitting a username to initiate a session, you can also use email=EMAILADDRESS in step 4 if those are unique for your end-users. For details see Generating authentication tokens for end-users via the API.