Implementing the API connector in your code

The local authentication API allows the use of local accounts via a REST API. Specifically, it enables an application to request an OpenAthens session-initiator URL for a local account by making a simple REST call to the authentication API. This will store a local account and associated attributes and permission sets and enable management via the administration UI.

This page offers guidance and code samples to help you integrate the relevant API calls into your systems to start OpenAthens sessions for users you have authenticated.

Prerequisites

- Familiarity with your own systems and how you will be authenticating users
- Experience of using an API programatically
- The connection ID and URI from the administration interface.

You will also need to support TLS v1.2 or above as connections using the older protocols will be rejected.

Method

Authentication

Authentication to the API is performed via an API key in the same way as other OpenAthens REST API requests. E.g:

```
Authorization: OAApiKey d0143c3f-383a-40d2-9594-00a7fc3bc15b
```

API keys can be generated via the Management menu of the OpenAthens administrator area. See: API keys.

Given what this key can be used for, you must ensure that your code does not reveal this key to the outside world

Base URL

All requests to the API use a consistent base URL:

```
https://login.openathens.net/api/v1/{domain}
```

As we are only interested in the authentication function here, we can include more in the base we will use:

```
https://login.openathens.net/api/v1/{domain}/organisation/{organisationID}/local-auth/session
```

Where {domain} is your OpenAthens customer domain and {organisationID} is your OpenAthens organisation ID

Your specific connection URI is displayed in the connection details in the administration interface and may have been passed to you by your library colleague.

Requesting a session initiator URL (POST)

Once you have authenticated your user, obtain a session initiator URL:

```
Content-type: application/vnd.eduserv.iam.auth.localAccountSessionRequest+json
POST /api/v1/{domain}/organisation/{oid}/local-auth/session HTTP/1.0

 {
   "connectionID" : "123",
   "uniqueUserIdentifier" : "asdf-fgfdgew321234",
   "displayName": "John Smith",
   "returnUrl" : "https://example.org/post-login",
   "attributes" : {
      "firstName" : "John",
      "lastName" : "Smith",
      "emailAddress" : "john.smith@example.org",
      "permissionSets" : [
         "example#default",
         "example#staff"
      ]
   }
}
```

The following properties may be passed in the request:

Property	Required	Description
connectionID	yes	Unique identifier for the local authentication system connection.
uniqueUserIdentifier	yes	A unique identifier for the end-user account in the local authentication system. This must be unique to each end-user, persistent between logins, and should ideally be pseudonymous and unique to the user for all time.
displayName	yes	A human-readable display name for the account holder. Appears in account lists and the audit trail.
returnUrl	yes	**A URL in your application** that the user will be returned to after the OpenAthens session initiator URL is visited.
attributes	only when marked as required in attribute mappings in UI	A set of additional attributes for the account holder that can be mapped via the UI. This is also where you would put permission set identifiers if mapping that way.

The response to the session initiator request will be a `application/vnd.eduserv.iam.auth.accountSessionInitiator+json` object:

```
Content-type: application/vnd.eduserv.iam.auth.accountSessionInitiator+json

 {
   expiry: "2015-09-22T13:57:31",
   sessionInitiatorUrl: "https://login.openthens.net/local/sso?
t=4534jkl54jk3h45k34jk4l35j3kl54j54kl35jk4j53klj435klj34kl5jkl
}
```

The following properties are passed in the response:

Property	Description
expiry	A timestamp indicating when this initiator URL will expire.
sessionInitiatorUrl	The URL that should be used to initiate an OpenAthens session.

The request may return one of the following HTTP response codes:

HTTP code	Description
200	The request was valid and an initiator URL was returned as above.
400	The request was invalid (usually due to missing mandatory parameters).
403	The API key is invalid, or The account has been suspended or banned.

500	Internal error in API.

Using a session-initiator URL

Once a session initiator URL is obtained, it may be used by a calling application to establish a users session. The application should initiate a 302 redirect to this URL. This will cause the OpenAthens authentication point to establish a session for the user based on the local account established in the initial URL request. The OpenAthens authentication point will then respond with a 302 redirect back to the URL supplied as the `returnUrl` parameter in the original request. The process is not user-interactive and the user will not see a visible UI during this request unless the initiator URL is corrupt and un-parsable, in which case the `returnUrl` becomes unavailable and errors are displayed.

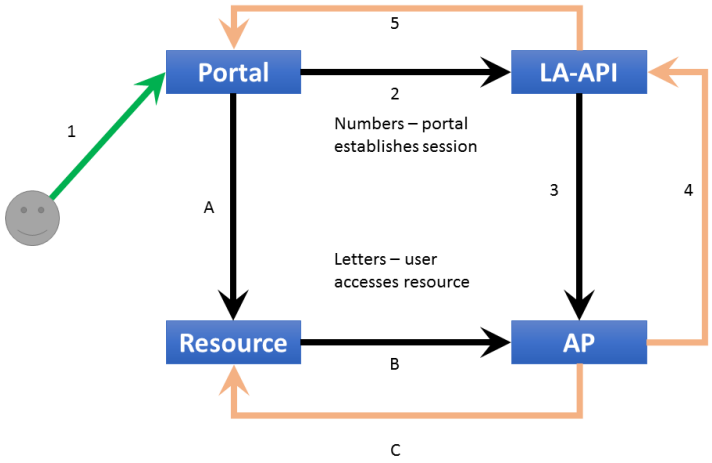
It is essential that the application does not alter the session initiator URL in any way, or attempt to infer any information from it.

When the user is then passed back to your application via the `returnUrl`, it will contain an additional `status` parameter to indicate whether the request was successful. The location specified by the return URL should be set up to expect this parameter and it may optionally be used by the calling application to take action depending on whether the operation was successful or otherwise. In fact, aside from token expiry, there are almost no reasons why a failure of the session initiator would occur .

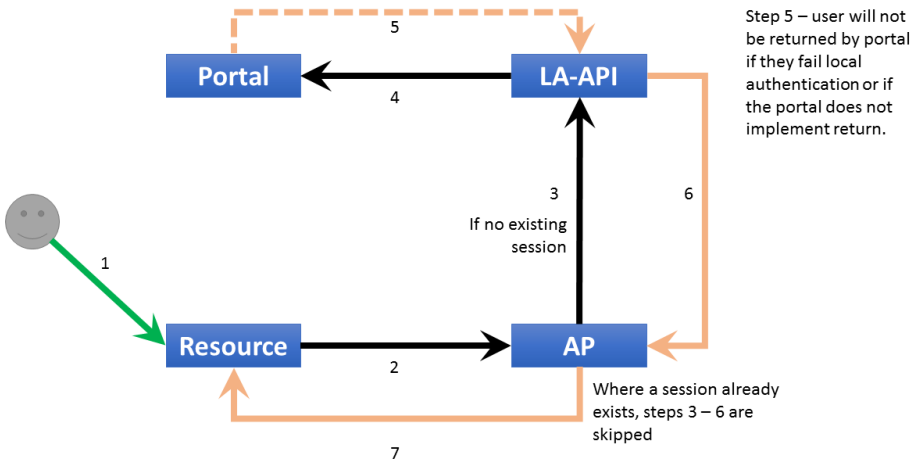
Status	Description
Success	The OpenAthens session was established successfully.
TokenExpired	The session initiator token in the request has expired (tokens are valid for 60 seconds after they are issued).
SessionFailure	The OpenAthens system was unable to establish the session (non-specfic error).

Authentication point callback request

The previous sections were focused on a local application establishing a session with the OpenAthens authentication point, where the user starts at the local application.E.g:



This next section covers what happens when the user attempts access to a service provider without an existing OpenAthens session. In such a case the OpenAthens authentication point is 'calling back' to the local application to perform a login on-demand - this is actually the normal mode of operation. It requires a little more work on your part, but provides your users with a way to get authenticated when they go to a resource first. The user journey instead goes like this:



The connection configuration in the administration site contains a field for a 'callback URL'. When a user searches and selects their organisation through the OpenAthens authentication point the user will be sent to this URL to login and then be returned to the authentication point to complete the round-trip to the service provider.

The OpenAthens authentication point will send a request to the configured callback URL with the following query-string parameters:

Parameter	Description
returnData	An opaque, signed packet generated by OpenAthens authentication point to encapsulate the SP request data so the the user can complete the round-trip to the SP. The local application MUST NOT alter or try to interpret the contents of this packet as it is private to OpenAthens authentication point and the format may change over time.

Responding to a OpenAthens authentication point callback request

To respond to a OpenAthens authentication point callback, the application receiving the callback request should authenticate the user and then perform a session initiator URL request (as described above), but replace the 'returnUrl' parameter with the 'returnData' parameter passed in the callback. The API will then generate an initiator URL that will respond back to the requesting SP instead of the returnUrl provided by the LA application.

Example request:

```
Content-type: application/vnd.eduserv.iam.auth.localAccountSessionRequest+json
POST /api/v1/{domain}/organisation/{oid}/local-auth/session HTTP/1.0

 {
   connectionID: "123",
   uniqueUserIdentifier: "asdf-fgfdgew321234",
   displayName: "John Smith",
   returnData: "dfgjkdfjg45kljy45kljtkljhkjekhltrjhk6elhjrklhjhklhjklh54jhklhjtrkhl54hehj90j"
   attributes: {
      firstName: "John",
      lastName: "Smith",
      emailAddress: "john.smith@example.org",
      permissionSets: [
         "example#default",
         "example#staff"
      ]
   }
}
```

Example response:

```
Content-type: application/vnd.eduserv.iam.auth.accountSessionInitiator+json

 {
   expiry: "2015-09-22T13:57:31",
   sessionInitiatorUrl: "https://login.openthens.net/local/sso?
t=4534jkl54jk3h45k34jk4l35j3kl54j54kl35jk4j53klj435klj34kl5jkl
}
```

Again this is not user-interacting and the process will be invisible to the user apart from the login functions of the resource and your portal or application.

Where in the user-flow should I invoke an OpenAthens session?

From the point of view of accessing resources: you will want to have this configured to act on a callback request as that will start an OpenAthens session at the point where it becomes necessary.

If your end-users log in to access your application or library page, you may like to initiate the session as part of that login for a smoother user experience.

Pseudonymous?

Pseudonymous identifiers are recommended for the unique user attribute to avoid potential problems with data protection legislation as that identifier will live on for a time in the audit trail after other mapped attributes are cleared.

The service desk can only support the API itself, not your systems. If you need expert consultancy please contact your account manager in the first instance.