Table of Contents

Setting Up Oauth Protection

If your application exposes at least one service (REST-ful or SOAP) that needs to be protected by the OAuth2.0 protocol, you need to deploy a special web application that will manage customers and issue access and refresh tokens. To deploy such an application, you will need to do the following:

1. Go to the Edit menu of the Configuration Tool
2. Select the entry with the name “SERVICE PROTECTION”

Note that the deployment of the dynamic feature should only be done once.

Callers' Workflow

This section describes how a third party (client) should call an Aware IM service protected by OAuth.

1. Before the client can call protected services, he must register himself with the OAUth server (this is done only once). To register, the client has to call this URL:

```
http(s)://ServerName:ServerPort/CXF/forms/registerClient.jsp
```

This should display a form where a client has to enter some mandatory and some optional values. The only value that requires explanation is “redirect URI”. This defines the URL that the OAuth server should call after creating an authorization token and returning to the client’s application, for example

```
http://ClientServerName/SomePath
```

1. When the form is submitted a reply is returned with the unique client id and secret. Client should store these values and use them when calling services
2. When calling a service to start a new authentication process the client has to go to this URL:

```
http(s)://ServerName:ServerPort/CXF/services/oauth/authorize?client_id=...&response_type=code&redirect_uri=...&scope=...
```

(redirect_uri is optional – if not specified, values registered for the client will be used) (if a scope is a general scope, provide scope as BusinessSpaceName_general)

1. The above URL call should return an “authorization code by calling the “redirect_uri” and supplying it with the “code” parameter. This code can then be exchanged for access token. To get access token the client has to call this URL using POST and supply the received code:

```
http(s)://ServerName:ServerPort/CXF/services/oauth/token?grant_type=authorization_code&code=...&client_id=...&client_secret=...&redirect_uri=...
```

1. HTTP headers of the request should be:
    1. `Accept: application/json"`
    2. `Content-Type: application/x-www-form-urlencoded`

1. Access token is returned as JSON string that looks like this:

```
{"tokenKey":"...","tokenType:"Bearer","refreshToken":...,"expiresIn":3600,"i
ssuedAt":-1,"issuer":null,"encedToken":null,"parameters":{},
"approvedScope":"general"}
```

1. The client should extract tokenKey and refreshToken (if he wants to refresh tokens). The client can then call Aware IM service with the provided token like so:

```
http://ServerName:ServerPort/CXF/services/serviceCall/api?serviceParameters
```

1. The following HTTP headers must be provided with the call:
    1. `Authorization=[Bearer accessToken]`
    2. `AW_DOMAIN=Name of business space`
    3. `AW_SERVICE=Name of Aware IM service`

Refresh Tokens

Refresh tokens are returned along with access token in a JSON string as part of the access token request:

```
{"tokenKey":"...","tokenType:"Bearer","refreshToken":...,"expiresIn":3600,"i
ssuedAt":-1,"issuer":null,"encedToken":null,"parameters":
```

A refresh token can then be exchanged for a new access token once the original access token expires with the following call (refresh token is specified in the “refresh_token” parameter

```
http://ServerName:ServerPort/CXF/services/oauth/token?grant_type=refresh_tok
en&refresh_token=...&client_id=...&client_secret=...&redirect_uri=...
```

The same reply as for the access token request should be received with the new access token and new refresh token.

Scopes

1. Clients must specify scope(s) in the authorization request. Requested scopes are separated by a space symbol (&scope=scope1%20scope2)
2. All requested scopes must be registered for the client. If no scopes are requested in the call, registered scopes are used.
3. The category of the called service must match one the approved scopes (approved by the end user)
4. Example of a valid service category:

BSPrefix_scope1_scope2_scope3 (provided that scope3 is not “general”)

Client Credentials Flow

The instructions above are for the “authorization” workflow. To use “client credentials workflow“ do the following:

1. In this flow the client just needs to call the AccessTokenService with a particular grant type. The client does not need to call the authorization service as the first step.
2. The first step should be calling AccessTokenService but instead of the authorization_code grant type you specify client_credentials grant type:

instead of this:

```
http://ServerName:ServerPort/CXF/services/oauth/token?grant_type=authorization_code&code=...&client_id=...&client_secret=...&redirect_uri=...
```

the client ashould call this:

```
http://ServerName:ServerPort/CXF/services/oauth/token?grant_type=client_credentials&client_id=...&client_secret=...
```

Summary

Your service documentation needs to convey to the callers of the service that before calling the service, they need to do the following:

1. Obtain an access token
2. include the access token in the Authorization Bearer HTTP header, for example (

   ```
   Authorization: Bearer <access token>
   ```

).
3. The name of the business space must be in the HTTP header called AW-DOMAIN, for example

   ```
   AW-DOMAIN:CRM
   ```
4. The name of the service to call must be in the header called AW-SERVICE for a REST-ful service (for example,

   ```
   AW-SERVICE: AddCustomer
   ```

) and AW-SERVICE-SOAP for a SOAP service - for example

   ```
   AW-SERVICE-SOAP: AddCustomer
   ```

   .