Trust Association Interceptor

Overview

Overview

MobileFirst Platform Foundation provides a Java library to facilitate the authentication of external resources.
The Java library is provided as a .jar file (**mfp-java-token-validator-8.0.0.jar**).

This tutorial will show how to protect a simple Java Servlet, `api/protected`, using a scope `ViewProtectedResource`.

Prerequesite:

- Make sure to read the Using the MobileFirst Server to authenticate external resources (../) tutorial.
- Understanding of the MobileFirst Platform Foundation security framework (../../).

Configuring the external service

For your external service to accept the access token, you must add a validation library to your service, which must be able to validate the token with either online or offline validation.

You can use one of two libraries for this purpose:

- `com.ibm.imf.oauth.common_1.0.0.jar` - Java lib
- The `node.js` library, which can be downloaded through `npm`.

Still need location on where I can get these files

External service configuration - Using Java: servlet filter

Server setup

1. Add the `com.ibm.imf.oauth.common_1.0.0.jar` file to your server under this path: `usr/extension/lib`
2. Add the `OAuthTai-1.0.mf` file to the path `usr/extension/lib/features`

Web.xml setup

Add a security constraint and a security role to the `web.xml` file of your external server, as shown below:

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>RESTServlet</web-resource-name>
    <url-pattern>/api/protected</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>TAIUserRole</role-name>
  </auth-constraint>
</security-constraint>

<security-role id="SecurityRole_TAIUserRole"; >
  <description>This is the role that MFP OAuthTAI uses to protect the resource, and it is required to be m
apped to 'All Authenticated in Application' in WAS and 'ALL_AUTHENTICATED_USERS' in Liberty</descr
iption>
  <role-name>TAIUserRole</role-name>
</security-role>
```

server.xml setup

You must modify the external server `server.xml` file to your external resource.

- Configure the feature manager to include these features:

```
<featureManager>
    <feature>jsp-2.2</feature>
    <feature>appSecurity-2.0</feature>
    <feature>usr:OAuthTai-8.0</feature>
    <feature>servlet-3.0</feature>
    <feature>jndi-1.0</feature>
</featureManager>
```

- In your application, add a security role:

```
<application id="REST-Server" location="REST-Server.war" name="REST-Server">
  <application-bnd>
    <security-role name="TAIUserRole">
      <special-subject type="ALL_AUTHENTICATED_USERS"/>
    </security-role>
  </application-bnd>
</application>
```

- Configure OAuthTAI, this is where URLs are set to be protected:

```
<usr_OAuthTAI id="myOAuthTAI" authorizationURL="http://localhost:9080/mfp/api" clientId="ExternalRes
ource" clientSecret="password" cacheSize="500">
      <securityConstraint httpMethods="GET POST" scope="ViewProtectedResource" securedURLs="/
REST-Server/api/protected"></securityConstraint>
</usr_OAuthTAI>
```

`authorizationURL`: Now that the validation/introspection is done against the MFP server on-line, you must notify the TAI what is the URL of the Authorization Server. This an either be your MFP Server, or some external AZ Server such as Datapower. In most cases, this will be the URL of your MFP Server: http://localhost:9080/mfp/api

`clientID`: The Resource server must be a registered confidential client, to learn how to register a confidential client

`clientSecret`: The Resource server must be a registered confidential client, to learn how to register a confidential client

`cacheSize`: The TAI has the optional cache, which can hold tokens as keys and introspection data as values, so that a token that comes in the request from the client won't need to be introspected again in a short time interval. The default size is 50,000 tokens.

`scope`: The Resource server must authenticate against the scope(s) that were defined with the clientID.

Reporting analytics Need more clarification on this

The **TAI** library is capable of reporting analytic events to IBM MobileFirst Platform Operational Analytics. To do this, you must configure the Resource Server that contains the **TAI** with your Analytics URL and credentials.

The MobileFirst Operational Analytics server is protected by basic authentication. When you installed this server, you configured the data entry point and basic authentication credentials.

> **Note:** Visit the knowledge center to learn more about installation of the Operational Analytics server.

To configure the **Resource Server**, you must provide the Analytics credentials, specifically the URL to the data entry point, the user name, and password. These properties are set with the following property names:

```
- "imf.analytics.url"
- "imf.analytics.username"
- "imf.analytics.password"
```

For example: If your Resource Server is running on Liberty, you must configure these properties by using JNDI. You can do this by adding entries to your `server.xml` file.

```
<jndiEntry jndiName="imf.analytics.username" value="admin"/>;
```

After these properties are set, the **TAI** will post its events to MobileFirst Operational Analytics.

Sample usage

1. Make sure to update the confidential client and secret values in the MobileFirst Operations Console.
2. Deploy either of the security checks: **UserLogin (../../user-authentication/security-check/)** or **PinCodeAttempts (../../credentials-validation/security-check/)**.
3. Register the matching application.
4. Map the `accessRestricted` scope to the security check.
5. Update the client application to make the `WLResourceRequest` to your servlet URL.