OAuth2 Token Endpoint and Servlet Filter Documentation

Introduction

This project implements OAuth2 Token Endpoint and Servlet Filter which enables adding OAuth2 based token authentication to your servlets. Current Token Endpoint supports 'Client Credentials' grant type only.

Modules

- **oauth2-server**: OAuth2 token endpoint web application
- **oauth2-filter**: Servlet filter which implements OAuth2 based authentication
- **oauth2-common**: Java library contaning implementations of functionalities common to both **oauth2-serve** and **oauth2-filter**.

Installation

Database setup

Current implementation only supports MySQL as the storage. You can find the databae schema(db-script.sql) at *$ROOT/modules/oauth2-common/src/main/resources*.

Add a resource to your Tomcat context.xml file located in $TOMCAT_ROOT/conf. Resource configuration will look like following.

```
<Resource name="jdbc/oauthStore" auth="Container" type="javax.sql.DataSource"
          maxActive="20" maxIdle="5" maxWait="10000"
          username="oauth2user" password="oauth2password"
driverClassName="com.mysql.jdbc.Driver"
          validationQuery="SELECT 1" testOnBorrow="true" testWhileIdle="true"
          timeBetweenEvictionRunsMillis="30000"
minEvictableIdleTimeMillis="30000"
          url="jdbc:mysql://localhost:8889/oauth"/>
```

Sample context.xml can be found inside *$ROOT/modules/oauth2-common/src/main/resources* directory of the source.

OAuth2 Server

You just need to copy the WAR file in to Tomcat webapps directory. You don’t need to do any configuration changes if you created data source resource with the name **jdbc/oauthStore**. If your resource name is different from that you need to change the value of the **ds-name** initial parameter in the web.xml.

OAuth2 Filter

You need to add following dependencies to your web apps pom.xml.

```
<dependency>
        <groupId>edu.indiana.d2i.htrc.oauth2</groupId>
        <artifactId>common</artifactId>
        <version>0.1-SNAPSHOT</version>
</dependency>
<dependency>
        <groupId>edu.indiana.d2i.htrc.oauth2</groupId>
        <artifactId>filter</artifactId>
        <version>0.1-SNAPSHOT</version>
</dependency>
```

This will make sure that all the dependencies required to OAuth2 Filter will get added to your web apps *lib* directory. *Make sure to check whether there is mysql jdbc connector jar is in your webapps lib directory*.

You need to have a data source resource configured in your Tomcat’s context.xml as described above and your filter configuration in web.xml will look like following.

```
<filter>
    <filter-name>oauth2-filter</filter-name>
    <filter-class>edu.indiana.d2i.htrc.oauth2.filter.OAuth2Filter</filter-class>
    <init-param>
        <param-name>ds-name</param-name>
        <param-value>jdbc/oauthStore</param-value>
    </init-param>
</filter>

<filter-mapping>
    <filter-name>oauth2-filter</filter-name>
    <url-pattern>/res</url-pattern>
</filter-mapping>

<resource-ref>
  <description>db conn</description>
  <res-ref-name>jdbc/oauthStore</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
</resource-ref>
```

Make sure to use correct **ds-name** initial parameter value which refers to data source resource created.

Sample Client

```
import org.apache.amber.oauth2.client.OAuthClient;
import org.apache.amber.oauth2.client.URLConnectionClient;
import org.apache.amber.oauth2.client.request.OAuthClientRequest;
import org.apache.amber.oauth2.client.response.OAuthAccessTokenResponse;
import org.apache.amber.oauth2.common.exception.OAuthProblemException;
import org.apache.amber.oauth2.common.exception.OAuthSystemException;
import org.apache.amber.oauth2.common.message.types.GrantType;
import org.apache.amber.oauth2.common.utils.OAuthUtils;

import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLConnection;

public class TestClient {
    public static void main(String[] args) throws OAuthSystemException,
OAuthProblemException, IOException {

        // We should not add grant_type=client_credentials param to URL
manually.
        // But current Amber implmentation doesn't automatically add that. So
this is required.
        OAuthClientRequest request = OAuthClientRequest
            .tokenLocation("http://localhost:8080/oauth2/token?
grant_type=client_credentials")
            .setGrantType(GrantType.CLIENT_CREDENTIALS)
            .setClientId("milinda")
            .setClientSecret("test123")
            .buildQueryMessage();

        OAuthClient oAuthClient = new OAuthClient(new URLConnectionClient());

        OAuthAccessTokenResponse response = oAuthClient.accessToken(request);

        // We can get the token by calling response.getAccessToken().
        // When accessing the resource you need to set the token as a HTTP
header.

        URL url = new URL("http://localhost:8080/oauth2/res");
        URLConnection c = url.openConnection();
        c.addRequestProperty(Common.HEADER_AUTHORIZATION,
response.getAccessToken());

        if (c instanceof HttpURLConnection) {
            HttpURLConnection httpURLConnection = (HttpURLConnection)c;
            httpURLConnection.setRequestMethod("GET");

            InputStream inputStream = null;
            if (httpURLConnection.getResponseCode() == 400) {
                inputStream = httpURLConnection.getErrorStream();
            } else {
                inputStream = httpURLConnection.getInputStream();
            }
            String responseBody = OAuthUtils.saveStreamAsString(inputStream);
            System.out.println(responseBody);
        }
    }
}
```

You should have the following dependencies in client pom.xml. org.apache.amber oauth2-common 0.22-incubating-SNAPSHOT

```
<dependency>
        <groupId>org.apache.amber</groupId>
        <artifactId>oauth2-client</artifactId>
        <version>0.22-incubating-SNAPSHOT</version>
</dependency>

<dependency>
        <groupId>org.apache.amber</groupId>
        <artifactId>oauth2-authzserver</artifactId>
        <version>0.22-incubating-SNAPSHOT</version>
</dependency>

<dependency>
        <groupId>org.apache.amber</groupId>
        <artifactId>oauth2-httpclient4</artifactId>
        <version>0.22-incubating-SNAPSHOT</version>
</dependency>
```

TODO: Needs to provide set of minimum required jars to write a client.

TODO

- Add user name to servlet request when authentication is successful.
- Log un-successful authentications at the OAuth2 filte
- OAuth2 server admin console.
- OAuth2 scope support for generating tokens.