GEMA

IWA

Benutzerleitfaden:
Authentifizierung am IWA-Webservice
am Beispiel eines JAVA-DemoClients

Inhaltsverzeichnis

1 Vorbemerkung und Abgrenzung

Die folgenden Abschnitte beschreiben die Einbindung des IWA-Webservices inkl. des Ablaufs der Authentifizierung beim ZBV-System (Zentrale Benutzerverwaltung) der GEMA.
Hierfür werden allgemeine Kenntnisse über Webservices und das HTTP Protokoll vorausgesetzt und daher nicht näher beschrieben.

Nach einer allgemeinen Ablaufdarstellung des Authentifizierungsvorgangs in IWA wird die Implementierung der Authentifizierung und die Anbindung des Webservices an einem JAVA-Beispiel vorgestellt.

2 Allgemeiner Ablauf der IWA-Webservice-Authentifizierung

Um den Webservice von IWA nutzen zu können, ist eine Authentifizierung beim ZBV-/SingleSignOn-System der GEMA notwendig. Dies und der Aufruf des Webservices passieren in folgenden Schritten (siehe auch Abbildung 1):

1) Aufruf der GEMA-ZBV-URL und Übergabe der Anmeldeparameter
2) Empfang eines Cookies mit einem Authentifizierungstoken
3) Aufrufen des Webservices und Übergabe des Authentifizierungstoken

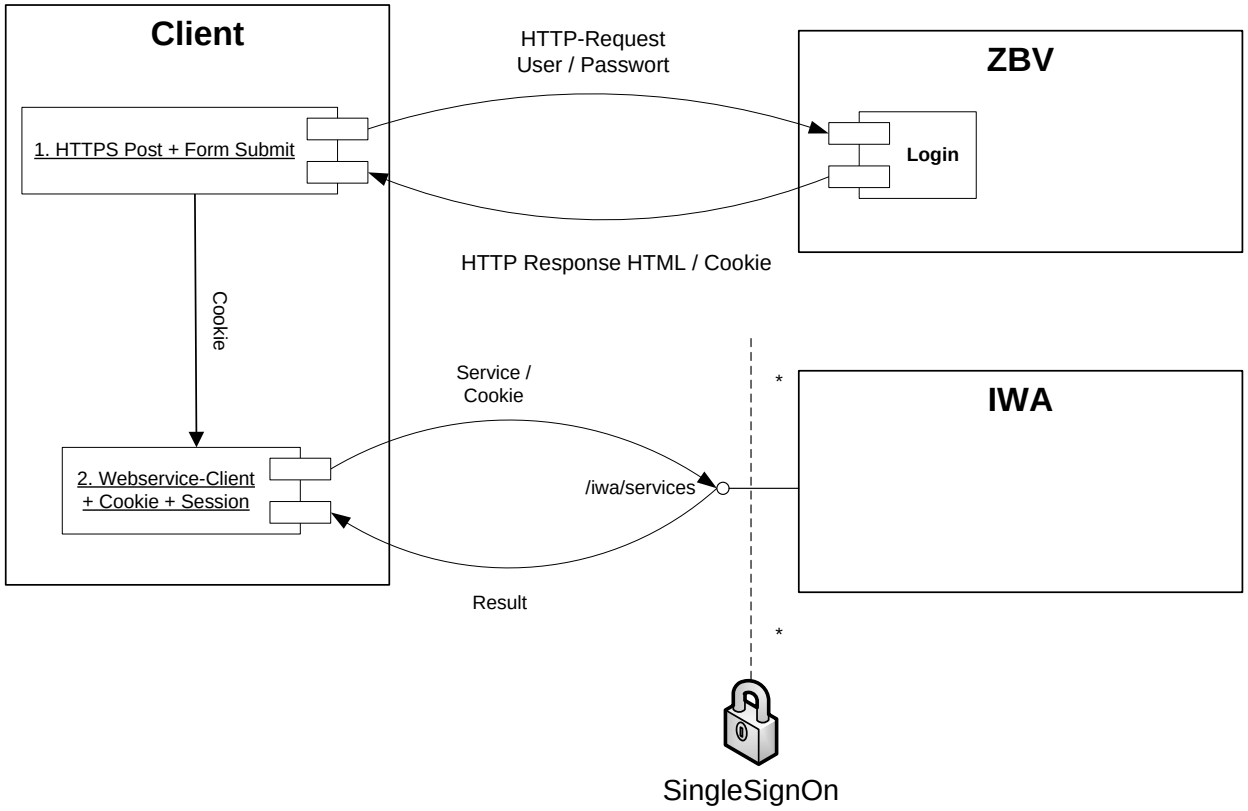

Abbildung 1: Schematische Darstellung des Authentifizierungsvorgangs

Zu Schritt 1)
Als erster Schritt ist eine Anmeldung am ZBV-System der GEMA notwendig, wofür ein Formular mit den Login-Daten an ZBV übermittelt werden muss. Über eine Post-Methode wird der Form-Submit an die URL **https://www.gema.de/zbv/login** übermittelt. Als Aufrufparameter werden dabei Benutzername, Passwort, sowie die eigentliche Aktion (Anmelden und Login) übergeben.

Zu Schritt 2)
Sollten User und Kennwort korrekt sein, liefert das ZBV-System Cookies zurück, die (unter anderem) den Authentifizierungstoken beinhalten.

Zu Schritt 3)

Dieses Cookie mit dem Authentifizierungstoken muss mit jedem Request des Webservices an IWA übergeben werden. Damit es vom Webservice Client aber tatsächlich übertragen wird, muss es entweder bei jedem Aufruf mitgegeben oder in einer clientseitigen Session gehalten werden (siehe Client-Beispiel im nächsten Kapitel).

Hinweis:
Sollte bei einem Aufruf kein Token übergeben werden (oder: sollte der Token bereits abgelaufen sein), wird als HTTP-Response-Code „Unauthorized“ (HTTP Code 401) zurückzugeben. Bei einem gültigen Token wird stattdessen das angeforderte Result geliefert.
Sollten das Token zwar Gültigkeit haben, der Benutzer jedoch für den Dienst nicht freigeschaltet worden sein, wird der Http Response Code „forbidden“ (HTTP Code 403) zurückgegeben.

3 Beschreibung am Beispiel eines Java-Clients

Die Anbindung des IWA-Webservices (inkl. ZBV-Authentifizierung) soll im Folgenden an einem kurzen JAVA-Beispiel demonstriert werden. Vorausgesetzt werden dafür Kenntnisse über JAVA, Apache HttpClient und Axis.
In den folgenden Absätzen werden die wichtigsten Variablen und Befehle der Demo-Anwendung erklärt. Der vollständige, zusammenhängende Quellcode findet sich im Anhang.

3.1 Voraussetzungen und zu importierende Pakete

Für die Realisierung wird clientseitig der HttpClient von Apache und der Webservice Client von Axis in der Version 1.4 benötigt:

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.Header;
import org.apache.commons.httpclient.NameValuePair;
import org.apache.commons.httpclient.URI;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.axis.transport.http.HTTPConstants;
```

Außerdem brauchen wir den von Axis generierten Client-Stub:

```
import iwaws.client.webservice.IwawsService;
import iwaws.client.webservice.IwawsServiceService;
import iwaws.client.webservice.IwawsServiceServiceLocator;
import iwaws.client.webservice.IwawsSoapBindingStub;
import java.net.URL;
```

3.2 Schritt 1: Aufruf der ZBV und Übergabe der Anmeldeparameter

Um sich beim ZBV-System der GEMA anzumelden, sind Benutzername und Passwort notwendig. Sie werden an ZBV übergeben und auf ihre Korrektheit geprüft.

```
String username;
String password;
```

Zurückgeliefert wird – bei korrekter Anmeldung – ein Cookie, das den notwendigen Authentifizierungstoken enthält. Dieses Token wird bei erfolgreicher ZBV-Anmeldung (siehe folgender Absatz) gesetzt und später im Rahmen des Webservice-Requests mit übergeben.

```
String token = null;
```

Die folgenden Zeilen zeigen den eigentlich Aufruf der GEMA-ZBV und den darauf folgenden Submit des Formulars. Zunächst wird dabei die URI auf die richtige Adresse gesetzt: *https://www.gema.de/zentrale-benutzerverwaltung-der-gema-online-services/login.html*.
Im Anschluss daran wird mit dem `NameValuePair[] data` das Formular zusammengesetzt, das an ZBV

übergeben werden muss. Es setzt sich zusammen aus dem Nutzernamen, dem Passwort und den „Aktionen“ die durch den Submit des Formulars ausgeführt werden sollen (Anmelden und Login).

```
PostMethod postMethod = new PostMethod();
postMethod.setURI(new URI("https://www.gema.de/zbv/login ", false));
NameValuePair[] data = new NameValuePair[] {
new NameValuePair("tx_crowd_pi1[user]", username),
        new NameValuePair("tx_crowd_pi1[pass]", password),
        new NameValuePair("tx_crowd_pi1[submit_button]", "Anmelden"),
        new NameValuePair("tx_crowd_pi1[logintype]", "login")  };
```

Zum Abschluss des Authentifizierungsvorgangs kann das so zusammengestellte Formular („data“) per Post-Methode übergeben werden.

```
postMethod.setRequestBody(data);
client.executeMethod(postMethod);
```

3.3 Schritt 2: Empfang des ZBV-Cookies

Die so übergebenen Daten werden von ZBV überprüft. Mithilfe der Methode `getResponseHeader` kann nun ein StringArray mit Cookies erzeugt werden.
„`GetResponseHeader`“ liefert dabei zunächst den Header. Über „`getValue`“ erhält man alle Cookies, die in diesem Header enthalten sind (getrennt durch einen Strichpunkt). Das StringArray kann über einen Split nach dem Strichpunkt erzeugt werden.

```
Header header =
postMethod.getResponseHeader(HTTPConstants.HEADER_SET_COOKIE);
String[] cookies = header.getValue().split(";");
```

Als Authentifizierungstoken wird jedoch nur der SingleSignOn-Teil benötigt, so dass innerhalb des String-Arrays nach dem richtigen Cookie (mit „gema-sso“) zu suchen ist.

```
for (String cookie : cookies) {
      if (cookie.contains("gema-sso")) {
            token = cookie;
            break;
      }
}
```

3.4 Schritt 3: Aufruf des Webservices und Übergabe des Authentifizierungstoken

Nach dem Erhalt des Tokens kann der von Axis generierte Webservice-Stub von IWA eingebunden werden, der unter *„https://online.gema.de/iwa/services/iwaws“* aufzurufen ist.

```
IwawsServiceService service = new IwawsServiceServiceLocator();
IwawsService iwaService =
service.getiwaws(new URL("https://online.gema.de/iwa/services/iwaws"));
```

Hier ist zusätzlich zu beachten, dass der „MaintainSession“-Parameter des Webservices-Client-Stubs zu konfigurieren ist. Er ist auf „true“ zu setzen, damit das ZBV-Cookie von Axis bei jedem Serviceaufruf übertragen wird.

```
((IwawsSoapBindingStub) iwaService)._setProperty
                        ("javax.xml.rpc.session.maintain", Boolean.TRUE);
```

Danach kann das Token über `setProperty(HTTPConstants.HEADER_COOKIE)` gesetzt werden.

```
((IwawsSoapBindingStub) iwaService)._setProperty
                        (HTTPConstants.HEADER_COOKIE, token);
```

Das so gelieferte Token wird vom SingleSignOn-Service der GEMA überprüft. Sollte der Benutzer ein gültiges Token übergeben, aber für den Dienst nicht freigeschaltet sein, wird als Response „forbidden“ (HTTP Code 403) geliefert. Sollte der Benutzer zwar freigeschaltet, das Token aber abgelaufen sein, erfolgt stattdessen die Response „unauthorized“ (HTTP Code 401).

4 Anhang

4.1 Vollständiger Quellcode des Beispiel-JavaClients

```
package iwaws.client.main;

import iwaws.client.webservice.IwawsService;
import iwaws.client.webservice.IwawsServiceService;
import iwaws.client.webservice.IwawsServiceServiceLocator;
import iwaws.client.webservice.IwawsSoapBindingStub;
import org.apache.axis.transport.http.HTTPConstants;
import org.apache.commons.httpclient.Header;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.NameValuePair;
import org.apache.commons.httpclient.URI;
import org.apache.commons.httpclient.methods.PostMethod;

import java.net.URL;

public class UserGuideDemo {

public static void  main(String[] args) throws Exception {

//Username und Passwort für den GEMA-ZBV-Login
String username = args[0];
String password = args[1];

HttpClient client = new HttpClient();
PostMethod postMethod = new PostMethod();

/*Das "Token" wird nach erfolgreicher Anmeldung
  von ZBV als "Authentifizierungsmerkmal" zurückgeliefert
  und muss beim Aufruf des Webservice übergeben werden
*/
String token = null;

/*Authentifizierungsvorgang bei GEMA-ZVB-Login
*/
try {
//Aufruf der ZBV-Login-URL
    postMethod.setURI(new URI("https://www.gema.de/zbv/login", false));
    NameValuePair[] data = new NameValuePair[] {
    new NameValuePair("tx_crowd_pi1[user]", username),
    new NameValuePair("tx_crowd_pi1[pass]", password),
    new NameValuePair("tx_crowd_pi1[submit_button]", "Anmelden"),
    new NameValuePair("tx_crowd_pi1[logintype]", "login")
    };

//Submit des Authentifizierungsformulars mit Nutzername und Passwort
    postMethod.setRequestBody(data);
    client.executeMethod(postMethod);

// Response: Rückgabe des Authentifizierungstokens als Cookie
```

```
    Header header = postMethod.getResponseHeader(HTTPConstants.HEADER_SET_COOKIE);
    String[] cookies = header.getValue().split(";");
    for (String cookie : cookies) {
       if (cookie.contains("gema-sso")) {
              token = cookie;
              break;
       }
    }
} finally {
       postMethod.releaseConnection();
}

/*Aufruf des GEMA-Webservices
*/
IwawsServiceService service = new IwawsServiceServiceLocator();
IwawsService iwaService =
       service.getiwaws(new URL("https://online.gema.de/iwa/services/iwaws"));


/*Damit das Cookie von Axis tatsächlich bei jedem Serviceaufruf übertragen wird,
  muss die MaintainSession auf "true" gesetzt werden.
*/

((IwawsSoapBindingStub) iwaService)._setProperty("javax.xml.rpc.session.maintain",
Boolean.TRUE);

//Übergabe des Authentifizierungs-Tokens
 ((IwawsSoapBindingStub) iwaService)._setProperty(HTTPConstants.HEADER_COOKIE, token);

//Webservice aufrufen
String wsVersion = iwaService.getVersion();

    }
}
```