deegree Web Authentication and Security Service v.2.5

lat/lon GmbH
Aennchenstr. 19
53177 Bonn
Germany
Tel ++49 - 228 - 184 96-0
Fax ++49 - 228 - 184 96-29
info@lat-lon.de
www.lat-lon.de

Dept. of Geography
Bonn University
Meckenheimer Allee 166
53115 Bonn

Tel. ++49 228 732098

Change log

Date	*Description*	*Author*
2007-01-11	Update using new formatting style	Markus Müller

Table of Contents

Index of Tables

Illustration Index

1 Introduction

deegree is a Java Framework offering the main building blocks for Spatial Data Infrastructures (SDIs). Its entire architecture is developed using standards of the Open Geospatial Consortium (OGC) and ISO Technical Committee 211 – Geographic information / Geoinformatics (ISO/TC 211). deegree encompasses OGC Web Services as well as clients. deegree is Free Software protected by the GNU Lesser General Public License (GNU LGPL) and is accessible at http://www.deegree.org.

deegree2 is the new release of deegree supporting a number of features that deegree1 was not able to handle. This documentation describes setup and configuration of the deegree Web Authentication Service (WAS) and/or a Web Security Service (WSS) according to the 1.0 version of the GDI-NRW specification.

In the deegree implementation, a WSS can currently do everything that a WAS can do, which is why there is only one documentation instead of two. In case the configuration of the two service types differs, it is noted. The WAS still has its use, in case the user wants to separate authentication from the actual secured service access.

Besides the WASS, deegree comprises a number of additional services and clients. A complete list of deegree components can be found at:

`http://www.lat-lon.de` → Products

Downloads of packaged deegree components can be found at:

`http://www.deegree.org` → Download

The web services of deegree are realized as Java modules controlled by one central servlet (the "dispatcher"). This servlet has to be deployed to the respective web server/servlet engine. Most of the common web servers support servlet technology, thus making deegree a universal product. The Apache-Tomcat 5.5 Servlet-Engine is recommended due to its widespread use and its status as an open-source product.

1.1 What both services can do

Both services can authenticate a user against a database (or just grant access to an anonymous user) and provide a session ID to avoid having to authenticate over and over again. To gain more insight into the authentication methods that can be used, see chapter three or the GDI-NRW specification.

1.2 What only the WSS can do

The WSS can process requests that include authentication information and a request for another OGC web service (DoService requests). These can be used to “secure” another service by hiding its real location and provide an authentication barrier. GetCapabilities responses can be processed to replace certain URLs with a facade URL.

The next chapter gives an overview about what you have to configure to set up a WAS and/or WSS securing another service. Chapter 3 gives a short rundown over the various authentication methods. Appendix A finally gives you complete configuration examples while appendix B provides some useful SQL scripts to create a user database.

2 Download / Installation

2.1 Prerequisites

For deegree2 Web Authentication and Security Service to run you need:

- Java (JRE or JSDK) version 1.5.x
- Tomcat 5.5.x
- an OGC web service to secure (may currently be WFS, WCS, WMS and CSW)

For installation of these components refer to the corresponding documentation at java.sun.com and tomcat.apache.org.

2.2 deegree WASS release

So far no demo download package is available for deegree WASS. Therefore you have to get the necessary files from the deegree CVS.

To set up a security service you need to set up the following components:

- a configuration for the WAS and/or the WSS (the WAS is optional)
- a database against which to authenticate users.
- a deegree2.jar (either precompiled or created from the classe in the CVS.

2.3 Setup of an OGC web service

To set up the OGC web service that needs to be secured, please consult the appropriate documentation. In order to make the resulting system reasonably secure, consider the following remarks, though.

If a GetCapabilities request is “piped” through a WSS, it replaces certain URLs in the response by a so called “facade URL” which can be provided by the user in the request. One should note that not all URLs will be replaced by the WSS, so if you have some informational fields containing the real location of the secured service that may be a security risk.

Furthermore - this should be obvious but noted - it makes much sense to configure your network and/or your service in a way that only the WSS can actually access it, so if you accidentally leave an occurrence of the location in the capabilities, it does not mean an actual security breach.

3 Basic Configuration

3.1 The WAS/WSS configuration documents

The configuration documents for a WAS and a WSS are actually capability documents with some extra information. You may already be familiar with this concept from other OGC web services implemented in deegree. The capabilities part of the configuration is straightforward, to learn how to do it just look into resources/wass/was/example/deegree/example_was_capabilities.xml file (resp. wss), or look in appendix A where you can find them as well. You just include the operations that you wish your service to support. Please be aware that certain operations are mandatory: the DescribeUser and the GetCapabilities operations are mandatory for both the WAS and the WSS, the DoService operation is mandatory for the WSS. The SAML part of the WAS is not supported by deegree at the moment.

Particularly interesting is the Capability element in both files, where you can setup which authentication methods will be supported by the service. It should be noted that in the case of a WSS the specified authentication methods will be accepted by both the GetSession and the DoService operations, so it is not possible to configure a WSS to accept the password method only for a GetSession operation and just the session method for the DoService method. In order to set up this scenario you'll need to set up a separate WAS. To learn more about authentication methods, see the next chapter. Please note that the WSS has a subelement under Capability that is called SecuredServiceType, where you specify the type of the secured service (for example WFS).

The various deegree parameters may also look familiar to people with deegree experience. The DefaultOnlineResource element is substituted for all omitted OnlineResource elements in the rest of the document.

To specify which database is to be used as user authentication backend, one specifies a JDBCConnection element under the deegree parameters according to the `http://www.deegree.org/jdbc` namespace. An example can be found in both configuration examples of appendix A.

The AuthenticationServiceAddress element can be used to build a chain of cascading security servers, all authentication requests will then be forwarded to the WAS that is specified here.

A RequestTimeLimit element specifies how long (in seconds) a request may process until an exception occurs, the default value if omitted is 15 seconds.

A SessionLifetime element specifies how long a session lives (if it is not closed), the default value if omitted is 1200 seconds.

In a WSS, there is a SecuredServiceName element containing a OnLineResource that specifies the address of the service that shall be hidden/secured by the WSS. All incoming DoService requests will use this service.

3.2 The user database

The database against which users are authenticated can generally be any database that you can use from within Java, which usually means that you should use a database with a JDBC driver.

To set up the database, you can use an SQL snippet for your database which can be found in scripts/sql/security/ in the deegree CVS or in appendix B. To avoid adding new users by hand, you can use the deegree user and rights management command line tool, whose documentation can be found in docs/dokumentation/security/drm_doku.pdf in the deegree CVS. Beware, some of the documentation you find in the security directory may be old/deprecated (especially the ones about the old WSS).

For an example how to integrate a PostgreSQL database in WASS, see the example configuration files' JDBCConnection element in the deegree parameters.

3.3 Tomcat Setup

To configure tomcat to serve as WAS/WSS, you have to configure two parts. The first part is the web.xml file of the context you are trying to set up. The easiest way to do so is to use the web.xml file in the deegree CVS (in webapps/deegree/WEB_INF/). In this file, set the init parameter “services” to include “was” and or “wss”, depending on your needs. For example, to start a WFS, WAS and a WSS, it should look as follows:

```
    <init-param>
      <param-name>services</param-name>
      <param-value>wfs,was,wss</param-value>
      <description>list of supported services, e.g.: wfs,wms (comma
separated)</description>
    </init-param>
```

Second, setup the init parameters specific to WAS/WSS to the desired values. The only thing of interest here is the was/wss.config variable which must point to the configuration file. Ignore the handler parameters if you're not a developer/code hacker.

```
    <!-- WAS INITIALIZING PARAMETERS -->
    <init-param>
      <param-name>was.config</param-name>
      <param-value>WEB-INF/conf/wass/was/example_was_capabilities.xml</param-value>
    </init-param>
```

```
<init-param>
  <param-name>was.handler</param-name>
  <param-value>org.deegree.enterprise.servlet.WASSHandler</param-value>
</init-param>

<!-- WSS INITIALIZING PARAMETERS -->
<init-param>
  <param-name>wss.config</param-name>
  <param-value>WEB-INF/conf/wass/wss/example_wss_capabilities.xml</param-value>
</init-param>
<init-param>
  <param-name>wss.handler</param-name>
  <param-value>org.deegree.enterprise.servlet.WASSHandler</param-value>
</init-param>
```

If you experiment a lot with the deegree web applications, it may be favorable to set up ant to build deegree, which enables you to use the deploy, undeploy and redeploy targets in our build.xml. Some quick hints to what you have to do:

- get all required and “optional” libraries for deegree (which you probably already have)
- copy them into libs/optional
- edit build.properties to suit your needs (tomcat location/port/username etc.)
- get the ant-contrib library from `http://ant-contrib.sourceforge.net`

4 Authentication methods

Authentication methods are what the name suggests. They are currently used for authentication in order to obtain a session using a GetSession request and for authentication in order to obtain access to a secured service using a DoService request.

The GDI-NRW standard specifies four different authentication methods, and labels them with a URN as follows:

`urn:x-gdi-nrw:authnMethod:1.0:password` – authentication method that uses a username/password pair for authentication. Username and password are separated by colons: joe,insecure

`urn:x-gdi-nrw:authnMethod:1.0:was` – authentication method involving a SAMLResponse object, currently not supported by deegree.

`urn:x-gdi-nrw:authnMethod:1.0:anonymous` – authentication method providing nothing for authentication.

`urn:x-gdi-nrw:authnMethod:1.0:session` – authentication method providing a previously obtained session ID for authentication.

Appendix A: Example configuration files

Example configuration for WAS:

```
<?xml version="1.0" encoding="UTF-8"?>
<was:WAS_Capabilities xmlns:was="http://www.gdi-nrw.org/was"
   xmlns:authn="http://www.gdi-nrw.org/authentication"
   xmlns="http://www.opengis.net/ows"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:deegreewas="http://www.deegree.org/was" version="1.1">
   <deegreewas:deegreeParam>
      <!--
         The DefaultOnlineResource will be used if a required OnlineResource is
         not defined
      -->
      <deegreewas:DefaultOnlineResource
         xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
         xlink:href="http://127.0.0.1:8080/deegree/services" />
      <!-- maximum time for the execution of a request until an exception of time-
exceed is thrown.
         optional; default 15 seconds -->
      <deegreewas:RequestTimeLimit>15</deegreewas:RequestTimeLimit>
      <deegreewas:AuthenticationServiceAddress>
         <!--
            mandatory element with online resource for accessing a WAS to
authenticate users
         -->
      <deegreewas:OnlineResource
            xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
            xlink:href="http://127.0.0.1:8081/deegree/was" />
      </deegreewas:AuthenticationServiceAddress>
      <JDBCConnection xmlns="http://www.deegree.org/jdbc">
     <Driver>org.postgresql.Driver</Driver>
     <Url>jdbc:postgresql://10.19.1.129:5432/wssusers</Url>
     <User>rutger</User>
     <Password>bla</Password>
     <SecurityConstraints/>
     <Encoding>iso-8859-1</Encoding>
   </JDBCConnection>
   </deegreewas:deegreeParam>

   <ServiceIdentification>
      <Title>lat-lon WAS</Title>
      <Abstract>
         A Web Security Service that secures the sectret WMS
      </Abstract>
      <Keywords>
         <Keyword>WAS</Keyword>
      </Keywords>
      <ServiceType>WAS</ServiceType>
      <ServiceTypeVersion>1.1</ServiceTypeVersion>
      <Fees>NONE</Fees>
      <AccessConstraints>NONE</AccessConstraints>
   </ServiceIdentification>
   <ServiceProvider>
      <ProviderName>lat-lon</ProviderName>
      <ProviderSite xlink:href="http://www.lat-lon.de" />
      <ServiceContact>
         <IndividualName>Andreas Poth</IndividualName>
         <PositionName>none</PositionName>
         <ContactInfo>
            <Phone>
               <Voice>+49 251 7474 432</Voice>
         </Phone>
         <Address>
            <DeliveryPoint>
               Aennchenstr. 19
            </DeliveryPoint>
            <City>Bonn</City>
            <AdministrativeArea>NRW</AdministrativeArea>
            <PostalCode>53177</PostalCode>
            <Country>Germany</Country>
            <ElectronicMailAddress>
               poth@lat-lon.de
            </ElectronicMailAddress>
         </Address>
         <OnlineResource xlink:href="http://www.lat-lon.de" />
      </ContactInfo>
   </ServiceContact>
</ServiceProvider>
<OperationsMetadata>
   <Operation name="GetCapabilities">
      <DCP>
         <HTTP>
            <Get xlink:href="http://www.lat-lon.de/WAS" />
         </HTTP>
      </DCP>
      <Parameter name="Format">
         <Value>text/xml</Value>
      </Parameter>
   </Operation>
   <Operation name="GetSession">
      <DCP>
         <HTTP>
            <Get xlink:href="http://www.lat-lon.de/WAS?" />
            <Post xlink:href="http://www.lat-lon.de/WAS" />
         </HTTP>
      </DCP>
      <Parameter name="Format">
         <Value>text/xml</Value>
      </Parameter>
   </Operation>
   <Operation name="CloseSession">
      <DCP>
         <HTTP>
            <Get xlink:href="http://www.lat-lon.de/WAS?" />
         </HTTP>
      </DCP>
      <Parameter name="Format">
         <Value>text/xml</Value>
      </Parameter>
   </Operation>
   <Operation name="DescribeUser">
      <DCP>
         <HTTP>
            <Get
               xlink:href="http://secure.service.com:8080/WSS?" />
            <Post
               xlink:href="http://secure.service.com:8080/WSS" />
         </HTTP>
      </DCP>
      <Parameter name="Format">
         <Value>text/xml</Value>
      </Parameter>
   </Operation>
   <!--   Currently not supported by deegree -->
   <!--
      <Operation name="GetSAMLResponse">
      <DCP>
      <HTTP>
      <Get xlink:href="http://www.lat-lon.de/WAS?"/>
      <Post xlink:href="http://www.lat-lon.de/WAS"/>
```

```
         </HTTP>
         </DCP>
         </Operation>
      -->
   </OperationsMetadata>
   <was:Capability>
      <was:SupportedAuthenticationMethodList>
         <!--authentication method #1-->
         <authn:SupportedAuthenticationMethod>
            <authn:AuthenticationMethod
               id="urn:x-gdi-nrw:authnMethod:1.0:password" />
         </authn:SupportedAuthenticationMethod>
         <!--authentication method #2-->
         <authn:SupportedAuthenticationMethod>
            <authn:AuthenticationMethod
               id="urn:x-gdi-nrw:authnMethod:1.0:session" />
         </authn:SupportedAuthenticationMethod>
      </was:SupportedAuthenticationMethodList>
   </was:Capability>
</was:WAS_Capabilities>

Example configuration for WSS:

<?xml version="1.0" encoding="UTF-8"?>
<wss:WSS_Capabilities version="1.0" updateSequence="0"
   xmlns:authn="http://www.gdi-nrw.org/authentication"
   xmlns="http://www.opengis.net/ows"
   xmlns:wss="http://www.gdi-nrw.org/wss"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:deegreewss="http://www.deegree.org/wss">
   <deegreewss:deegreeParam>
      <!--
         The DefaultOnlineResource will be used if a required OnlineResource is
         not defined
      -->
      <deegreewss:DefaultOnlineResource
         xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
         xlink:href="http://127.0.0.1:8081/deegree/ogcwebservice" />
      <!-- maximum time for the execution of a request until an exception of time-
exceed is thrown.
         optional; default 1200 seconds -->
      <deegreewss:RequestTimeLimit>1200</deegreewss:RequestTimeLimit>
      <deegreewss:SecuredServiceAddress>
         <!--
            mandatory element with online resource for accessing the capabilities of
the hidden service
         -->
         <deegreewss:OnlineResource
            xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
            xlink:href="http://127.0.0.1:8081/deegree/ogcwebservice" />
      </deegreewss:SecuredServiceAddress>
      <deegreewss:AuthenticationServiceAddress>
         <!--
            mandatory element with online resource for accessing a WAS to
authenticate users
         -->
         <deegreewss:OnlineResource
            xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
            xlink:href="http://127.0.0.1:8081/deegree/was" />
      </deegreewss:AuthenticationServiceAddress>

      <JDBCConnection xmlns="http://www.deegree.org/jdbc">
     <Driver>org.postgresql.Driver</Driver>
     <Url>jdbc:postgresql://10.19.1.129:5432/wssusers</Url>
     <User>rutger</User>
     <Password>bla</Password>
     <SecurityConstraints/>
     <Encoding>iso-8859-1</Encoding>
```

```
</JDBCConnection>
</deegreewss:deegreeParam>
<ServiceIdentification>
   <Title>Deegree WSS</Title>
   <Abstract>
      A Web Security Service that secures a restricted WFS
   </Abstract>
   <Keywords>
      <Keyword>WSS</Keyword>
   </Keywords>
   <ServiceType>WSS</ServiceType>
   <ServiceTypeVersion>1.0</ServiceTypeVersion>
   <Fees>Free Access</Fees>
   <AccessConstraints>None</AccessConstraints>
</ServiceIdentification>
<ServiceProvider>
   <ProviderName>My Compagny</ProviderName>
   <ProviderSite xlink:href="http://secure.service.com" />
   <ServiceContact>
      <IndividualName>Andreas Poth</IndividualName>
      <PositionName>Research Associate</PositionName>
      <ContactInfo>
         <Phone>
            <Voice>+49 251 8333084</Voice>
         </Phone>
         <Address>
            <DeliveryPoint>Elmstreet 12</DeliveryPoint>
            <City>Small town</City>
            <AdministrativeArea>NRW</AdministrativeArea>
            <PostalCode>12345</PostalCode>
            <Country>Germany</Country>
            <ElectronicMailAddress>
               poth@lat-lon.de
            </ElectronicMailAddress>
         </Address>
         <OnlineResource
            xlink:href="http://secure.service.com/~poth" />
      </ContactInfo>
   </ServiceContact>
</ServiceProvider>
<OperationsMetadata>
   <Operation name="GetCapabilities">
      <DCP>
         <HTTP>
            <Get
               xlink:href="http://secure.service.com:8080/WSS?" />
            <Post
               xlink:href="http://secure.service.com:8080/WSS" />
         </HTTP>
      </DCP>
      <Parameter name="Format">
         <Value>text/xml</Value>
      </Parameter>
   </Operation>
   <Operation name="GetSession">
      <DCP>
         <HTTP>
            <Get
               xlink:href="http://secure.service.com:8080/WSS?" />
            <Post
               xlink:href="http://secure.service.com:8080/WSS" />
         </HTTP>
      </DCP>
      <Parameter name="Format">
         <Value>text/xml</Value>
      </Parameter>
   </Operation>
   <Operation name="CloseSession">
      <DCP>
```

```
            <HTTP>
               <Get
                  xlink:href="http://secure.service.com:8080/WSS?" />
               <Post
                  xlink:href="http://secure.service.com:8080/WSS" />
            </HTTP>
         </DCP>
         <Parameter name="Format">
            <Value>text/xml</Value>
         </Parameter>
      </Operation>
      <Operation name="DoService">
         <DCP>
            <HTTP>
               <Get
                  xlink:href="http://secure.service.com:8080/WSS?" />
               <Post
                  xlink:href="http://secure.service.com:8080/WSS" />
            </HTTP>
         </DCP>
         <Parameter name="Format">
            <Value>text/xml</Value>
         </Parameter>
      </Operation>
      <Operation name="DescribeUser">
         <DCP>
            <HTTP>
               <Get
                  xlink:href="http://secure.service.com:8080/WSS?" />
               <Post
                  xlink:href="http://secure.service.com:8080/WSS" />
            </HTTP>
         </DCP>
         <Parameter name="Format">
            <Value>text/xml</Value>
         </Parameter>
      </Operation>
   </OperationsMetadata>
   <wss:Capability>
      <wss:SecuredServiceType>WFS</wss:SecuredServiceType>
      <wss:SupportedAuthenticationMethodList>
         <!--authentication method #1 password-->
         <authn:SupportedAuthenticationMethod>
            <authn:AuthenticationMethod
               id="urn:x-gdi-nrw:authnMethod:1.0:password" />
         </authn:SupportedAuthenticationMethod>
         <!--authentication method #2 session-->
         <authn:SupportedAuthenticationMethod>
            <authn:AuthenticationMethod
               id="urn:x-gdi-nrw:authnMethod:1.0:session" />
         </authn:SupportedAuthenticationMethod>
         <!--authentication method #3 anonymous-->
         <authn:SupportedAuthenticationMethod>
            <authn:AuthenticationMethod
               id="urn:x-gdi-nrw:authnMethod:1.0:anonymous" />
         </authn:SupportedAuthenticationMethod>
      </wss:SupportedAuthenticationMethodList>
   </wss:Capability>
</wss:WSS_Capabilities>
```

Appendix B: SQL scripts

For MS Access:

```
DROP TABLE SEC_GROUPS;
CREATE TABLE "HUIS"."SEC_GROUPS" (
"ID" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_JT_GROUPS_GROUPS;
CREATE TABLE "HUIS"."SEC_JT_GROUPS_GROUPS" (
"FK_GROUPS_MEMBER" NUMBER(10),
"FK_GROUPS" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_JT_GROUPS_ROLES;
CREATE TABLE "HUIS"."SEC_JT_GROUPS_ROLES" (
"FK_GROUPS" NUMBER(10),
"FK_ROLES" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_JT_ROLES_PRIVILEGES;
CREATE TABLE "HUIS"."SEC_JT_ROLES_PRIVILEGES" (
"FK_ROLES" NUMBER(10),
"FK_PRIVILEGES" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_JT_ROLES_SECOBJECTS;
CREATE TABLE "HUIS"."SEC_JT_ROLES_SECOBJECTS" (
"FK_ROLES" NUMBER(10),
"FK_SECURABLE_OBJECTS" NUMBER(10),
"FK_RIGHTS" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_JT_USERS_GROUPS;
CREATE TABLE "HUIS"."SEC_JT_USERS_GROUPS" (
"FK_USERS" NUMBER(10),
"FK_GROUPS" NUMBER(10)
)
```

```
TABLESPACE "HUIS";


DROP TABLE SEC_JT_USERS_ROLES;
CREATE TABLE "HUIS"."SEC_JT_USERS_ROLES" (
"FK_USERS" NUMBER(10),
"FK_ROLES" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_PRIVILEGES;
CREATE TABLE "HUIS"."SEC_PRIVILEGES" (
"ID" NUMBER(10),
"NAME" VARCHAR(50)
)
TABLESPACE "HUIS";


DROP TABLE SEC_RIGHTS;
CREATE TABLE "HUIS"."SEC_RIGHTS" (
"ID" NUMBER(10),
"NAME" VARCHAR(50)
)
TABLESPACE "HUIS";


DROP TABLE SEC_ROLES;
CREATE TABLE "HUIS"."SEC_ROLES" (
"ID" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_SECURABLE_OBJECTS;
CREATE TABLE "HUIS"."SEC_SECURABLE_OBJECTS" (
"ID" NUMBER(10),
"NAME" VARCHAR(255),
"TITLE" VARCHAR(255)
)
TABLESPACE "HUIS";


DROP TABLE SEC_SECURED_OBJECT_TYPES;
CREATE TABLE "HUIS"."SEC_SECURED_OBJECT_TYPES" (
"ID" NUMBER(10),
"NAME" VARCHAR(50)
)
TABLESPACE "HUIS";


DROP TABLE SEC_SECURED_OBJECTS;
CREATE TABLE "HUIS"."SEC_SECURED_OBJECTS" (
"ID" NUMBER(10),
"FK_SECURED_OBJECT_TYPES" NUMBER(10)
)
TABLESPACE "HUIS";


DROP TABLE SEC_USERS;
CREATE TABLE "HUIS"."SEC_USERS" (
"ID" NUMBER(10),
"FIRSTNAME" VARCHAR(255),
"LASTNAME" VARCHAR(255),
"EMAIL" VARCHAR(255)
)
TABLESPACE "HUIS";

INSERT INTO "HUIS"."SEC_PRIVILEGES" ("ID", "NAME") VALUES (1, 'write');
INSERT INTO "HUIS"."SEC_PRIVILEGES" ("ID", "NAME") VALUES (2, 'adduser');
INSERT INTO "HUIS"."SEC_PRIVILEGES" ("ID", "NAME") VALUES (3, 'addgroup');
INSERT INTO "HUIS"."SEC_PRIVILEGES" ("ID", "NAME") VALUES (4, 'addrole');
INSERT INTO "HUIS"."SEC_PRIVILEGES" ("ID", "NAME") VALUES (5,
'addservice');

INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (1, 'access');
INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (2, 'query');
INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (3, 'delete');
INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (4, 'insert');
INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (5, 'execute');
INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (6, 'update');
INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (7, 'view');
INSERT INTO "HUIS"."SEC_RIGHTS" ("ID", "NAME") VALUES (8, 'grant');

INSERT INTO "HUIS"."SEC_SECURED_OBJECT_TYPES" ("ID", "NAME") VALUES (1,
'dataset');
INSERT INTO "HUIS"."SEC_SECURED_OBJECT_TYPES" ("ID", "NAME") VALUES (2,
'wms');
INSERT INTO "HUIS"."SEC_SECURED_OBJECT_TYPES" ("ID", "NAME") VALUES (3,
'wfs');


INSERT INTO "HUIS"."SEC_SECURABLE_OBJECTS" ("ID", "NAME", "TITLE") VALUES
(1, 'SEC_ADMIN', 'Security Admin User');
INSERT INTO "HUIS"."SEC_USERS" ("ID", "FIRSTNAME", "LASTNAME", "EMAIL")
VALUES (1, 'SEC_ADMIN', 'SEC_ADMIN', 'mschneider@lat-lon.de');

INSERT INTO "HUIS"."SEC_SECURABLE_OBJECTS" ("ID", "NAME", "TITLE") VALUES
(2, 'SEC_ADMIN', 'Security Admin Group');
INSERT INTO "HUIS"."SEC_GROUPS" ("ID") VALUES (2);

INSERT INTO "HUIS"."SEC_SECURABLE_OBJECTS" ("ID", "NAME", "TITLE") VALUES
(3, 'SEC_ADMIN', 'Security Admin Role');
INSERT INTO "HUIS"."SEC_ROLES" ("ID") VALUES (3);
INSERT INTO "HUIS"."SEC_JT_GROUPS_ROLES" ("FK_GROUPS", "FK_ROLES") VALUES
(2, 3);
INSERT INTO "HUIS"."SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES")
VALUES (3, 1);
INSERT INTO "HUIS"."SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES")
VALUES (3, 2);
INSERT INTO "HUIS"."SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES")
VALUES (3, 3);
INSERT INTO "HUIS"."SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES")
VALUES (3, 4);
INSERT INTO "HUIS"."SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES")
VALUES (3, 5);

INSERT INTO "HUIS"."SEC_JT_ROLES_SECOBJECTS" ("FK_ROLES",
"FK_SECURABLE_OBJECTS", "FK_RIGHTS") VALUES (3, 2, 8);
INSERT INTO "HUIS"."SEC_JT_USERS_GROUPS" ("FK_USERS", "FK_GROUPS") VALUES
(1, 2);
COMMIT;
```

For Oracle:

```
DROP TABLE SEC_GROUPS;
CREATE TABLE "SEC_GROUPS" (
"ID" NUMBER(10)
);

DROP TABLE SEC_JT_GROUPS_GROUPS;
CREATE TABLE "SEC_JT_GROUPS_GROUPS" (
"FK_GROUPS_MEMBER" NUMBER(10),
"FK_GROUPS" NUMBER(10)
);

DROP TABLE SEC_JT_GROUPS_ROLES;
CREATE TABLE "SEC_JT_GROUPS_ROLES" (
"FK_GROUPS" NUMBER(10),
"FK_ROLES" NUMBER(10)
);

DROP TABLE SEC_JT_ROLES_PRIVILEGES;
CREATE TABLE "SEC_JT_ROLES_PRIVILEGES" (
"FK_ROLES" NUMBER(10),
"FK_PRIVILEGES" NUMBER(10),
"CONSTRAINTS" CLOB
);

DROP TABLE SEC_JT_ROLES_SECOBJECTS;
CREATE TABLE "SEC_JT_ROLES_SECOBJECTS" (
"FK_ROLES" NUMBER(10),
"FK_SECURABLE_OBJECTS" NUMBER(10),
"FK_RIGHTS" NUMBER(10),
"CONSTRAINTS" CLOB
```

```
);

DROP TABLE SEC_JT_USERS_GROUPS;
CREATE TABLE "SEC_JT_USERS_GROUPS" (
"FK_USERS" NUMBER(10),
"FK_GROUPS" NUMBER(10)
);

DROP TABLE SEC_JT_USERS_ROLES;
CREATE TABLE "SEC_JT_USERS_ROLES" (
"FK_USERS" NUMBER(10),
"FK_ROLES" NUMBER(10)
);

DROP TABLE SEC_PRIVILEGES;
CREATE TABLE "SEC_PRIVILEGES" (
"ID" NUMBER(10),
"NAME" VARCHAR(50)
);

DROP TABLE SEC_RIGHTS;
CREATE TABLE "SEC_RIGHTS" (
"ID" NUMBER(10),
"NAME" VARCHAR(50)
);

DROP TABLE SEC_ROLES;
CREATE TABLE "SEC_ROLES" (
"ID" NUMBER(10)
);

DROP TABLE SEC_SECURABLE_OBJECTS;
CREATE TABLE "SEC_SECURABLE_OBJECTS" (
"ID" NUMBER(10),
"NAME" VARCHAR(255),
"TITLE" VARCHAR(255)
);

DROP TABLE SEC_SECURED_OBJECT_TYPES;
CREATE TABLE "SEC_SECURED_OBJECT_TYPES" (
"ID" NUMBER(10),
"NAME" VARCHAR(50)
);

DROP TABLE SEC_SECURED_OBJECTS;
CREATE TABLE "SEC_SECURED_OBJECTS" (
"ID" NUMBER(10),
"FK_SECURED_OBJECT_TYPES" NUMBER(10)
);

DROP TABLE SEC_USERS;
CREATE TABLE "SEC_USERS" (
"ID" NUMBER(10),
"PASSWORD" VARCHAR(255),
"FIRSTNAME" VARCHAR(255),
"LASTNAME" VARCHAR(255),
"EMAIL" VARCHAR(255)
);

DELETE FROM SEC_JT_GROUPS_GROUPS;
DELETE FROM SEC_JT_GROUPS_ROLES;
DELETE FROM SEC_JT_ROLES_PRIVILEGES;
DELETE FROM SEC_JT_ROLES_SECOBJECTS;
DELETE FROM SEC_JT_USERS_GROUPS;
DELETE FROM SEC_JT_USERS_ROLES;
DELETE FROM SEC_PRIVILEGES;
DELETE FROM SEC_RIGHTS;
DELETE FROM SEC_ROLES;
DELETE FROM SEC_USERS;
DELETE FROM SEC_GROUPS;
DELETE FROM SEC_SECURED_OBJECT_TYPES;
DELETE FROM SEC_SECURED_OBJECTS;
DELETE FROM SEC_SECURABLE_OBJECTS;

INSERT INTO "SEC_PRIVILEGES" ("ID", "NAME") VALUES (1, 'write');
INSERT INTO "SEC_PRIVILEGES" ("ID", "NAME") VALUES (2, 'adduser');
INSERT INTO "SEC_PRIVILEGES" ("ID", "NAME") VALUES (3, 'addgroup');
INSERT INTO "SEC_PRIVILEGES" ("ID", "NAME") VALUES (4, 'addrole');
INSERT INTO "SEC_PRIVILEGES" ("ID", "NAME") VALUES (5, 'addobject');

INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (1, 'access');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (2, 'query');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (3, 'delete');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (4, 'insert');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (5, 'execute');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (6, 'update');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (7, 'view');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (8, 'grant');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (9, 'GetMap');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (10, 'GetFeatureInfo');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (11, 'GetLegendGraphic');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (12, 'GetScaleBar');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (13, 'GetFeature');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (14, 'DescribeFeatureType');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (15, 'GetCoverage');
INSERT INTO "SEC_RIGHTS" ("ID", "NAME") VALUES (16, 'DescribeCoverage');

INSERT INTO "SEC_SECURABLE_OBJECTS" ("ID", "NAME", "TITLE") VALUES (1,
'SEC_ADMIN', 'Security Admin User');
INSERT INTO "SEC_USERS" ("ID", "PASSWORD", "FIRSTNAME", "LASTNAME",
"EMAIL") VALUES (1, 'JOSE67', 'SEC_ADMIN', 'SEC_ADMIN', NULL);

INSERT INTO "SEC_SECURABLE_OBJECTS" ("ID", "NAME", "TITLE") VALUES (2,
'SEC_ADMIN', 'Security Admin Group');
INSERT INTO "SEC_GROUPS" ("ID") VALUES (2);
INSERT INTO "SEC_SECURABLE_OBJECTS" ("ID", "NAME", "TITLE") VALUES (3,
'SEC_ADMIN', 'Security Admin Role');
INSERT INTO "SEC_ROLES" ("ID") VALUES (3);

INSERT INTO "SEC_JT_GROUPS_ROLES" ("FK_GROUPS", "FK_ROLES") VALUES (2, 3);
INSERT INTO "SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES") VALUES
(3, 1);
INSERT INTO "SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES") VALUES
(3, 2);
INSERT INTO "SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES") VALUES
(3, 3);
INSERT INTO "SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES") VALUES
(3, 4);
INSERT INTO "SEC_JT_ROLES_PRIVILEGES" ("FK_ROLES", "FK_PRIVILEGES") VALUES
(3, 5);

INSERT INTO "SEC_JT_ROLES_SECOBJECTS" ("FK_ROLES", "FK_SECURABLE_OBJECTS",
"FK_RIGHTS") VALUES (3, 2, 8);
INSERT INTO "SEC_JT_USERS_GROUPS" ("FK_USERS", "FK_GROUPS") VALUES (1, 2);
```

For PostgreSQL:

```
DROP TABLE SEC_GROUPS;
CREATE TABLE SEC_GROUPS (
ID INTEGER
);

DROP TABLE SEC_JT_GROUPS_GROUPS;
CREATE TABLE SEC_JT_GROUPS_GROUPS (
FK_GROUPS_MEMBER INTEGER,
FK_GROUPS INTEGER
);

DROP TABLE SEC_JT_GROUPS_ROLES;
CREATE TABLE SEC_JT_GROUPS_ROLES (
FK_GROUPS INTEGER,
FK_ROLES INTEGER
);

DROP TABLE SEC_JT_ROLES_PRIVILEGES;
CREATE TABLE SEC_JT_ROLES_PRIVILEGES (
FK_ROLES INTEGER,
FK_PRIVILEGES INTEGER,
CONSTRAINTS TEXT
);

DROP TABLE SEC_JT_ROLES_SECOBJECTS;
CREATE TABLE SEC_JT_ROLES_SECOBJECTS (
FK_ROLES INTEGER,
FK_SECURABLE_OBJECTS INTEGER,
FK_RIGHTS INTEGER,
CONSTRAINTS TEXT
);

DROP TABLE SEC_JT_USERS_GROUPS;
CREATE TABLE SEC_JT_USERS_GROUPS (
FK_USERS INTEGER,
FK_GROUPS INTEGER
);

DROP TABLE SEC_JT_USERS_ROLES;
CREATE TABLE SEC_JT_USERS_ROLES (
FK_USERS INTEGER,
FK_ROLES INTEGER
);

DROP TABLE SEC_PRIVILEGES;
CREATE TABLE SEC_PRIVILEGES (
ID INTEGER,
NAME VARCHAR(50)
);

DROP TABLE SEC_RIGHTS;
CREATE TABLE SEC_RIGHTS (
ID INTEGER,
NAME VARCHAR(50)
);

DROP TABLE SEC_ROLES;
CREATE TABLE SEC_ROLES (
ID INTEGER
);

DROP TABLE SEC_SECURABLE_OBJECTS;
CREATE TABLE SEC_SECURABLE_OBJECTS (
ID INTEGER,
NAME VARCHAR(255),
TITLE VARCHAR(255)
);

DROP TABLE SEC_SECURED_OBJECT_TYPES;
CREATE TABLE SEC_SECURED_OBJECT_TYPES (
ID INTEGER,
NAME VARCHAR(50)
);

DROP TABLE SEC_SECURED_OBJECTS;
CREATE TABLE SEC_SECURED_OBJECTS (
ID INTEGER,
FK_SECURED_OBJECT_TYPES INTEGER
);
```

```
DROP TABLE SEC_USERS;
CREATE TABLE SEC_USERS (
ID INTEGER,
PASSWORD VARCHAR(255),
FIRSTNAME VARCHAR(255),
LASTNAME VARCHAR(255),
EMAIL VARCHAR(255)
);

DELETE FROM SEC_JT_GROUPS_GROUPS;
DELETE FROM SEC_JT_GROUPS_ROLES;
DELETE FROM SEC_JT_ROLES_PRIVILEGES;
DELETE FROM SEC_JT_ROLES_SECOBJECTS;
DELETE FROM SEC_JT_USERS_GROUPS;
DELETE FROM SEC_JT_USERS_ROLES;
DELETE FROM SEC_PRIVILEGES;
DELETE FROM SEC_RIGHTS;
DELETE FROM SEC_ROLES;
DELETE FROM SEC_USERS;
DELETE FROM SEC_GROUPS;
DELETE FROM SEC_SECURED_OBJECT_TYPES;
DELETE FROM SEC_SECURED_OBJECTS;
DELETE FROM SEC_SECURABLE_OBJECTS;

INSERT INTO SEC_PRIVILEGES (ID, NAME) VALUES (1, 'write');
INSERT INTO SEC_PRIVILEGES (ID, NAME) VALUES (2, 'adduser');
INSERT INTO SEC_PRIVILEGES (ID, NAME) VALUES (3, 'addgroup');
INSERT INTO SEC_PRIVILEGES (ID, NAME) VALUES (4, 'addrole');
INSERT INTO SEC_PRIVILEGES (ID, NAME) VALUES (5, 'addobject');

INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (1, 'access');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (2, 'query');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (3, 'delete');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (4, 'insert');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (5, 'execute');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (6, 'update');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (7, 'view');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (8, 'grant');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (9, 'GetMap');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (10, 'GetFeatureInfo');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (11, 'GetLegendGraphic');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (12, 'GetScaleBar');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (13, 'GetFeature');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (14, 'DescribeFeatureType');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (15, 'GetCoverage');
INSERT INTO SEC_RIGHTS (ID, NAME) VALUES (16, 'DescribeCoverage');

INSERT INTO SEC_SECURABLE_OBJECTS (ID, NAME, TITLE) VALUES (1, 'SEC_ADMIN',
'Security Admin User');
INSERT INTO SEC_USERS (ID, PASSWORD, FIRSTNAME, LASTNAME, EMAIL) VALUES (1,
'JOSE67', 'SEC_ADMIN', 'SEC_ADMIN', NULL);
```

```
INSERT INTO SEC_SECURABLE_OBJECTS (ID, NAME, TITLE) VALUES (2, 'SEC_ADMIN',
'Security Admin Group');
INSERT INTO SEC_GROUPS (ID) VALUES (2);

INSERT INTO SEC_SECURABLE_OBJECTS (ID, NAME, TITLE) VALUES (3, 'SEC_ADMIN',
'Security Admin Role');
INSERT INTO SEC_ROLES (ID) VALUES (3);

INSERT INTO SEC_JT_GROUPS_ROLES (FK_GROUPS, FK_ROLES) VALUES (2, 3);
INSERT INTO SEC_JT_ROLES_PRIVILEGES (FK_ROLES, FK_PRIVILEGES) VALUES (3,
1);
INSERT INTO SEC_JT_ROLES_PRIVILEGES (FK_ROLES, FK_PRIVILEGES) VALUES (3,
2);
INSERT INTO SEC_JT_ROLES_PRIVILEGES (FK_ROLES, FK_PRIVILEGES) VALUES (3,
3);
INSERT INTO SEC_JT_ROLES_PRIVILEGES (FK_ROLES, FK_PRIVILEGES) VALUES (3,
4);
INSERT INTO SEC_JT_ROLES_PRIVILEGES (FK_ROLES, FK_PRIVILEGES) VALUES (3,
5);

INSERT INTO SEC_JT_ROLES_SECOBJECTS (FK_ROLES, FK_SECURABLE_OBJECTS,
FK_RIGHTS) VALUES (3, 2, 8);
INSERT INTO SEC_JT_USERS_GROUPS (FK_USERS, FK_GROUPS) VALUES (1, 2);
```