

Installation

Table of contents

1 Introduction.....	2
2 Configure the datasource.....	2
3 Installation of the Flexible Jdbc Realm.....	2
4 Configure glassfish with the realm definition.....	2
5 Configure the application with a realm definition.....	4

1. Introduction

FlexibleJdbcRealm is a glassfish security realm and is deployed as an additional jar of the glassfish runtime.

2. Configure the datasource

To start off, a datasource must be configured for accessing the database with user and group information.

3. Installation of the Flexible Jdbc Realm

- Copy the FlexibleJdbcRealm jar to the glassfish lib directory.

4. Configure glassfish with the realm definition

- Add mapping of the security realm to the flexible JDBC realm login module to the login.conf of the glassfish domain you are using. For example:

```
PhotoXChangeRealm {
    org.wamblee.glassfish.auth.FlexibleJdbcLoginModule required;
};
```

- Configure the security realm in glassfish.
 - In admin console, go to Configuration/Security/Realms.
 - Add a new realm and use the class name org.wamblee.glassfish.auth.FlexibleJdbcRealm as the realm class.
 - Configure the realm as follows:

Property	Description
jaas.context	The name of the realm as in login.conf and web.xml.
sql.password	The sql prepared statement that returns the encoded password for a given user. The username is the single parameter of the realm.
sql.groups	The sql prepared statement which returns the groups based on the username.
datasource.jndi	The jndi name of the datasource.

<p>password.digest</p>	<p>The digest method of the password. The value of this property is an encoding as supported by <code>MessageDigest.getInstance(String)</code>. The special value <code>PLAIN</code> must be used to indicate that no encoding will be used. Use for instance <code>MD5</code> for MD5 digests.</p> <p>Note: In version 0.1, the value <code>MD5HEX</code> was used for MD5, this must now be simply <code>MD5</code></p> <p>Note: In version 0.2, the name of this property was <code>password.encoding</code>. In later versions, the digest method and encoding are separately configurable.</p>
<p>password.encoding</p>	<p>Optional property for the encoding of the digest result into a string. Two encoding methods are supported:</p> <ul style="list-style-type: none"> • <code>HEX</code>: For hexadecimal encoding. • <code>BASE64</code>: For base64 encoding. <p>If this property is not set then the value <code>HEX</code> is used.</p>
<p>sql.seed</p>	<p>Optional property to define the query to retrieve the seed for a given user name. This is a query in prepared statement syntax with one parameter for the user name. If this property is not defined then the empty string will be used for the seed (which is identical to no seeding).</p>
<p>password.seed.format</p>	<p>Optional property that defined how the seed will be used together with the password. This is a string in <code>java.text.MessageFormat</code> format with the first parameter the password and the second parameter the seed. If this property is not set than the default format is <code>"{0}{1}"</code> meaning that the seed will be appended to the password.</p>
<p>assign-groups</p>	<p>Optional parameter with a comma-separated list of groups that any</p>

	authenticated user is assigned to.
--	------------------------------------

5. Configure the application with a realm definition

- Add realm definition to web.xml login-config element (realm-name), For example:

```
<login-config>
  <auth-method>FORM</auth-method>
  <realm-name>PhotoXChangeRealm</realm-name>
  <form-login-config>
    <form-login-page>/login.jsp</form-login-page>
    <form-error-page>/loginError.jsp</form-error-page>
  </form-login-config>
</login-config>
```

Note:

In glassfish V2 b58, `<role-name>*/role-name</role-name>` does not work and you should use the `assign-groups` property to assign any authenticated user to a specific group. That specific group should be used to represent any authenticated user instead of using `<role-name>*/role-name</role-name>`.