FEWS Web Services Tomcat Security

Table of Contents

Introduction

This section describes how the FEWS Web Services can be protected using the security mechanisms of the Tomcat application server.

In general it is recommended to handle authentication outside of tomcat. When running in a cloud enviroment, this can typically be done with an API gateway. In on premise enviroments a reverse proxy (like NGinx or Apache HTTPD) or a firewall/applicance is recommended. Using FEWS Web Services Security with Open ID Connect is natively supported by the Delft-FEWS Web Services since 2022.01.

In this section an example is given of using Basic Authentication as security measure using the default UserDatabaseRealm using a file based user store. Alternatively Tomcat has support for connecting to an LDAP server or using a JDBC connection to a database to access user accounts.

Tomcat 8 or 9 is recommended above Tomcat 7 for it's advanced security possibilities using the so called CredentialHandler

Basic Authentication with Tomcat

To configure Basic Authentication for the FEWS Web Services with theUserDatabaseRealm, the Tomcat application server has to be configured as described in the following steps. Since these steps are configured in the tomcat application server, they apply to all web applications running in that tomcat server.

In the conf directory of the tomcat installation a **web.xml** file is available. At the end of this file the following xml should be added (just before the closing web-app tag):

web.xml

```
    <security-role>
        <role-name>fewswebservices</role-name>
    </security-role>

    <security-constraint>
        <web-resource-collection>
            <web-resource-name>
                FEWS Web Services
            </web-resource-name>
            <url-pattern>/*</url-pattern>
        </web-resource-collection>
        <auth-constraint>
            <role-name>fewswebservices</role-name>
        </auth-constraint>
    </security-constraint>

    <login-config>
        <auth-method>BASIC</auth-method>
        <realm-name>FEWS Web Services</realm-name>
    </login-config>
```

This configuration will apply basic authentication to all FEWS Web Services.

Basic authentication should always be used with a secure HTTPS connection. This is usually done by adding a secure proxy in front of Tomcat like NGINX or the Apache Webserver. Alternatively Tomcat itself can be configured for using HTTPS connection.

User management with Tomcat

Tomcat has the concept of a Realm to store users, roles and passwords. For more information see Tomcat Realm implementations. In a standard tomcat installation, tomcat is configured with a file based user store that can be used with basic authentication called the UserDatabaseRealm. For more advanced implementations, please consult the tomcat documentation. In the conf directory of the Tomcat installation the **server.xml** file can be found where this realm is configured.

The UserDatabaseRealm uses the **tomcat-users.xml** file from the conf folder to store users, roles and passwords. By default the passwords are stored in plain text. It is strongly advised to use a hashing algorithm to prevent storing plain text passwords on the file system. Both Tomcat 7 and 8 support hashing algorithms. Tomcat 8 and higher use a CredentialHandler that has support for salt and iterations which is not possible with Tomcat 7.

User management with Tomcat 8 and 9

In the following **server.xml** file the **CredentialHandler** element was added to the UserDatabaseRealm. In this case the PBKDF2WithHmacSHA512 algorithm is configured with a keyLength of 256, a saltLength of 16 and 100000 iterations. These settings are a trade-off between performance and security. The exact values should be evaluated per use case.

server.xml

```
<Realm className="org.apache.catalina.realm.LockOutRealm">
   <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
          resourceName="UserDatabase">
      <CredentialHandler className="org.apache.catalina.realm.SecretKeyCredentialHandler"
                  algorithm="PBKDF2WithHmacSHA512"
                  iterations="100000"
                  keyLength="256"
                  saltLength="16"
      />
   </Realm>
</Realm>
```

Now tomcat has been configured, users can be added that are allowed access to the FEWS Web Services. The following is an example of a **tomcat-users.xml** file where a user (dummy_username) and a fewswebservices role has been added. All users with the role fewswebservices will get access to the FEWS Web Services. The file can be found in the conf directory the tomcat installation:

tomcat-users.xml

```
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">

<role rolename="fewswebservices"/>
<user username="dummy_username" password="dummy_password" roles="fewswebservices"/>
</tomcat-users>
```

Choose a strong password instead of the dummy_password used in this example!

In this example the password still has been set in plain text. To get the hashed version of the password, tomcat provides the digest tool in the bin folder of the tomcat installation. To generate a hashed version of the dummy_password password, the following command can be issued (on Windows, the command is available on Linux as well). Note that the algorithm, number of iterations, salt length and keyLength all are passed to the tool:

```
digest.bat -a "PBKDF2WithHmacSHA512" -i 100000 -s 16 -k 256 -h "org.apache.catalina.realm.
SecretKeyCredentialHandler" dummy_password
```

This will result in the following output with the original password, followed by a : and finally the hashed value of the password:

```
dummy_password:
91429c93e8b1d9462852770ea94d3cee$100000$48c94a74968e5a1b5df394a50c27effeb330553b66dc75d7840a9beb25a2ce90
```

The **tomcat-users.xml** file can now be updated with the hashed value, which will look as follows:

tomcat-users.xml

```
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">

<role rolename="fewswebservices"/>
<user username="dummy_username" password="
91429c93e8b1d9462852770ea94d3cee$100000$48c94a74968e5a1b5df394a50c27effeb330553b66dc75d7840a9beb25a2ce90"
roles="fewswebservices"/>
</tomcat-users>
```

Now when accessing the FEWS Web Services the user fews can access all webserver pages with the Test1234 password.

User management with Tomcat 7

To enable hashing a **digest** attribute has to be added to the UserDatabaseRealm with the hashing algorithm algorithm to be used.

In the following **server.xml** file the **digest** attribute was added to the UserDatabaseRealm and as digest algorithm **SHA-512** has been configured.

server.xml

```
  <Realm className="org.apache.catalina.realm.LockOutRealm">
        <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
               resourceName="UserDatabase" digest="SHA-512" />
      </Realm>
```

Now tomcat has been configured, users can be added that are allowed access to the FEWS Web Services. The following is an example of a **tomcat-users.xml** file where a user (dummy_username) and a fewswebservices role has been added. All users with the role fewswebservices will get access to the FEWS Web Services. The file can be found in the conf directory the tomcat installation:

tomcat-users.xml

```
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">

<role rolename="fewswebservices"/>
<user username="dummy_username" password="dummy_password" roles="fewswebservices"/>
</tomcat-users>
```

Choose a strong password instead of the dummy_password used in this example!

In this example the password still has been set in plain text. To get the hashed version of the password, tomcat provides the digest tool in the bin folder of the tomcat installation. To generate a hashed version of the Test1234 password, the following command can be issued (on Windows, the command is available on Linux as well):

```
digest.bat -s 0 -a SHA-512 dummy_password
```

This will result in the following output with the original password, followed by a : and finally the hashed value of the password:

```
dummy_password:
b43f1d28a3dbf30070bf1ae7c88ee2784047fc86d7be8620c8510debbd8555b3ef0b96376a4dd494ae0561580274bcf7a3069f5c0beceff6
3d1237a13d4d72b7
```

The **tomcat-users.xml** file can now be updated with the hashed value, which will look as follows:

tomcat-users.xml

```
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">

<role rolename="fewswebservices"/>
<user username="dummy_username" password="
b43f1d28a3dbf30070bf1ae7c88ee2784047fc86d7be8620c8510debbd8555b3ef0b96376a4dd494ae0561580274bcf7a3069f5c0beceff6
3d1237a13d4d72b7" roles="fewswebservices"/>
</tomcat-users>
```

Now when accessing the FEWS Web Services the user dummy_username can access all webserver pages with the dummy_password password.

Advanced access configuration

In the previous examples all FEWS Web Services are available for all users with the fewswebservices role. It is also possible to configure a more fine grained access to the different web services or even methods within the services by using different roles and different url-patterns in the security constraints. Each FEWS Web Service has its own url pattern that can be used. The different patterns per web service are as follows.

- FEWS PI Rest Web Services: **/rest/fewspiservice/***
- FEWS PI SOAP Web Services: **/fewspiservice**
- FEWS Digitale Delta Web Services: **/rest/digitaledelta/***
- FEWS WaterML Web Services: **/waterml**
- FEWS Umaquo Web Services: **/umaquo**
- FEWS WMS Web Services: **/wms**
- FEWS Schematic Status Display Web Services: **/ssd**

These patterns can be used in the web.xml configuration to specify more specific security-constraint elements.

Using these url patterns, it is possible to give access to specific services or resources by configuring dedicated groups. For example to allow only access to the FEWS PI Rest Web Services to a specific group of users, the following configuration changes can be made to the web.xml.

Define a new security-role **fewspirest.** Add a security-constraint using the url-pattern **/rest/fewspiservice/*** and add as auth-constraint both the **fewspirest** and **fewswebservices** roles. This results in the following web.xml.

web.xml

```
<security-role>
    <role-name>fewswebservices</role-name>
</security-role>
 <security-role>
    <role-name>fewspirest</role-name>
</security-role>

<security-constraint>
    <web-resource-collection>
        <web-resource-name>
            FEWS Web Services
        </web-resource-name>
        <url-pattern>/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
        <role-name>fewswebservices</role-name>
    </auth-constraint>
</security-constraint>

<security-constraint>
    <web-resource-collection>
        <web-resource-name>
            FEWS PI Rest Web Services
        </web-resource-name>
        <url-pattern>/rest/fewspiservice/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
        <role-name>fewspirest</role-name>
                <role-name>fewswebservices</role-name>
    </auth-constraint>
</security-constraint>

<login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>FEWS Web Services</realm-name>
</login-config>
</web-app>
```

In the following example the user "dummy_username_rest" has been assigned the role "fewpirest". The "dummy_username_rest" user will only have access to the FEW PI Rest service.

tomcat-users.xml

```
<role rolename="fewswebservices"/>
<role rolename="fewspirest"/>

<user username="dummy_username" roles="fewswebservices" password="
91429c93e8b1d9462852770ea94d3cee$100000$48c94a74968e5a1b5df394a50c27effeb330553b66dc75d7840a9beb25a2ce90" />
<user username="dummy_username_rest" roles="fewspirest" password="[PASSWORD_OF_USER_dummy_username_rest" />

</tomcat-users>
```

General recommendations

- Always inspect the Tomcat documentation on the latest security improvements.
- Take note that generating hashes of passwords on the machine where the passwords are stored can still keep references to the password in for example a history file. Take measurements to avoid the passwords from being logged. For example in Linux bash starting a command with a space, will prevent the command from being added to the history file.
- Always use strong passwords