Secure connection in Tomcat and Authentication for the Client application

1. Generate keystore file:

[path-to-tomcat]/bin>keytool -genkey -alias [NAME] -keyalg RSA -keystore [PATH]

Replace [NAME] by a name, [PATH] by the the file that store the key. For example:
[NAME] = tsp; [PATH] = C:\.keystore

You will be asked for some information such as name, password, etc. Just follow the instructions.
Assume that the password is: coapstsp

2. Enable HTTPS in Tomcat by modifying the [path-to-tomcat]/conf/server.xml:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
maxThreads="150" scheme="https" secure="true"
keystoreFile="[path-to]/.keystore" keystorePass="[your-key-password]"
clientAuth="false" sslProtocol="TLS" />
```

For example:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
maxThreads="150" scheme="https" secure="true"
keystoreFile="C:/.keystore" keystorePass=" coapstsp "
clientAuth="false" sslProtocol="TLS" />
```

3. Declare a new role and users in Tomcat by modifying the [path-to-tomcat]/conf/tomcat-users.xml:

```
<role rolename="[ROLE-NAME]"/>
<user username="[USERNAME]" password="[PASSWORD]" roles="[ROLE-NAME]"/>
```

Replace [ROLE-NAME], [USERNAME], [PASSWORD] by suitable text.

For example:

```
<role rolename="coaps-client"/>
<user username="chan" password="chan" roles="coaps-client"/>
<user username="mohamed" password="mohamed" roles="coaps-client"/>
<user username="sami" password="sami" roles="coaps-client"/>
```

4. In the application (star-PaaS-Client), add the following codes between the <webapp> tags, in the src/main/webapp/WEB-INF/web.xml:

```
 <security-constraint>
   <web-resource-collection>
     <web-resource-name>Wildcard means whole app requires authentication</web-resource-name>
     <url-pattern>/*</url-pattern>
     <http-method>GET</http-method>
     <http-method>POST</http-method>
   </web-resource-collection>
   <auth-constraint>
     <role-name>[ROLE-NAME]</role-name>
   </auth-constraint>
   <user-data-constraint>
     <transport-guarantee>NONE</transport-guarantee>
   </user-data-constraint>
 </security-constraint>
   <security-role>
         <description>security role assignment for Non-Existent Role – this prevents direct access to
JSPs</description>
         <role-name>[ROLENAME]</role-name>
 </security-role>
 <login-config>
   <auth-method>BASIC</auth-method>
 </login-config>
```

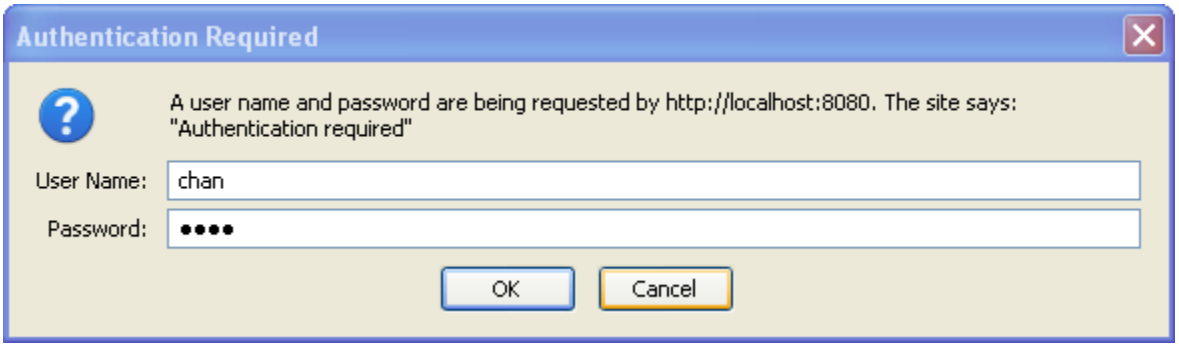
Replace [ROLE-NAME] by the name declared in the [path-to-tomcat]/conf/tomcat-users.xml

For example:

```
<security-constraint>
   <web-resource-collection>
     <web-resource-name>Wildcard means whole app requires authentication</web-resource-name>
     <url-pattern>/*</url-pattern>
     <http-method>GET</http-method>
     <http-method>POST</http-method>
   </web-resource-collection>
   <auth-constraint>
     <role-name>coaps-client</role-name>
   </auth-constraint>
   <user-data-constraint>
     <transport-guarantee>NONE</transport-guarantee>
   </user-data-constraint>
 </security-constraint>
   <security-role>
         <description>security role assignment for Non-Existent Role – this prevents direct access to
JSPs</description>
```

```
        <role-name>coaps-client</role-name>
</security-role>
<login-config>
  <auth-method>BASIC</auth-method>
</login-config>
```

5. Rebuild the Client by Maven install.
6. Open http://localhost:8080/client. The browser will ask about the username & password. Prompt the username/password of the role coaps-client declared in the [path-to-tomcat]/conf/tomcat-users.xml, such as chan/chan; mohamed/mohamed; etc.



7. Open https://localhost:8443/client. The browser will ask you about accessing to an un-trusted connection. This is because the keystore file is created locally without the certification from a trusted security company. Just click on “I understand the risks” -> Add Exceptions -> Confirm security exception, and type the username/password if asked.