Configuració autenticació bàsica a Canigó 3.1

A qui va dirigit

Aquest how-to va dirigit a tots aquells usuaris que vulguin afegir autenticació bàsica a una aplicació Canigó 3.1 REST.

Versió de Canigó

Els passos descrits en aquest document apliquen a la darrera versió del Framework Canigó 3.1.x.

Introducció

Configuració de l'autenticació bàsica a una aplicació Canigó Rest mitjançant web.xml o el servei de seguretat de Canigó que utilitza Spring Security.

Configuració autenticació bàsica a Canigó 3.1

Web.xml + Servidor web(Tomcat)

Es pot configurar la seguretat per a les peticions REST a través del web.xml utilitzant el tag **security-constraint**.
Amb Security Constraint s'especifica els recursos WEB a protegir (Patrones URL, Mètodes HTTP), restriccions d'autenticació (Roles)

web.xml

```
…
<security-constraint>
          <web-resource-collection>
               <web-resource-name>REST Services</web-resource-name>
               <url-pattern>/services/*</url-pattern>
               <http-method>GET</http-method>
               <http-method>PUT</http-method>
               <http-method>POST</http-method>
               <http-method>DELETE</http-method>
          </web-resource-collection>
          <auth-constraint>
              <role-name>administrator</role-name>
          </auth-constraint>
    </security-constraint>
…
```

En aquest exemple hem protegit qualsevol petició que compleixi la petició **/services/*** i realitzi un PUT, POST o DELETE. Els mètodes no afegits no queden protegits. Si es vol assegurar tots els mètodes per evitar qualsevol tipus de vulnerabilitat no s'ha de posar cap http-method.
També indiquem amb auth-constraint que només un usuari amb rol administrator té permís per a realitzar les accions protegides.

Amb el tag **login-config** es configura l'autenticació (es pot seleccionar BASIC, DIGEST, CLIENT o FORM).
En cas de voler tenir la flexibilitat de poder mostrar el nostre formulari d'accés s'ha de seleccionar FORM.

En aquest exemple per simplicitat utilitzarem BASIC.

web.xml

```
…
      <login-config>
        <auth-method>BASIC</auth-method>
        <realm-name>default</realm-name>
    </login-config>
…
```

Al Servidor web Tomcat hem d'afegir el rol administrator que demanem per autenticar, i els usuaris que pertanyen a aquest grup.
Aquesta configuració es realitza al fitxer tomcat-users.xml

Configuració autenticació bàsica a Canigó 3.1

```
…
<role rolename="administrator"/>
<user username="user" password="password" roles="administrator"/>
…
```

Configuració autenticació bàsica a Canigó 3.1

Spring Security

A les aplicacions REST també es pot utilitzar el servei Canigo-security de la mateixa manera que a les aplicacions JSF.

S'ha d'afegir la dependència al servei de seguretat:

```
…
<properties>
      <canigo.security>[1.1.0,1.2.0)</canigo.security>
…

    <dependency>
            <groupId>cat.gencat.ctti</groupId>
            <artifactId>canigo.security</artifactId>
            <version>${canigo.security}</version>
      </dependency>
…
```

Al fitxer web.xml configurem el filtre de seguretat de Spring

```
…
<filter>
      <filter-name>springSecurityFilterChain</filter-name>
      <filter-
class>org.springframework.web.filter.DelegatingFilterProxy</filter-
class>
</filter>
<filter-mapping>
      <filter-name>springSecurityFilterChain</filter-name>
      <url-pattern>/*</url-pattern>
</filter-mapping>
…
```

Es crea el fitxer app-custom-security.xml on es configura les url que s'han de protegir i el mètode d'autenticació. Per a l'exemple protegim totes les url que pengin de services i utilitzem el fitxer security.users.properties per als usuaris/rols

security.users.properties

```
user=userspassword,ROLE_USER,enabled
admin2=adminspassword2,ROLE_USER,ROLE_ADMIN,enabled</filter-mapping>
```

Configuració autenticació bàsica a Canigó 3.1

app-custom-security.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:security="http://www.springframework.org/schema/security"
      xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.1.xsd
            http://www.springframework.org/schema/security
http://www.springframework.org/schema/security/spring-security-
3.2.xsd">

      <!--  Secure patterns -->
      <security:http auto-config="true">
            <security:intercept-url pattern="/services/**"
access="ROLE_ADMIN"/>
      </security:http>

      <security:authentication-manager>
            <!-- In-Memory Authentication provider -->
            <security:authentication-provider>
                  <security:password-encoder hash="plaintext"/>
                  <security:user-service
properties="classpath:/config/props/security.users.properties"/>
            </security:authentication-provider>
      </security:authentication-manager>
</beans>
```

De la mateixa manera que hem utilitzat un fitxer de text pla com a authentication manager es pot configurar que s'autentiqui mitjançant GICAR o un altre mètode d'autenticació.