

CONFIGURACIÓN DE HTTPS

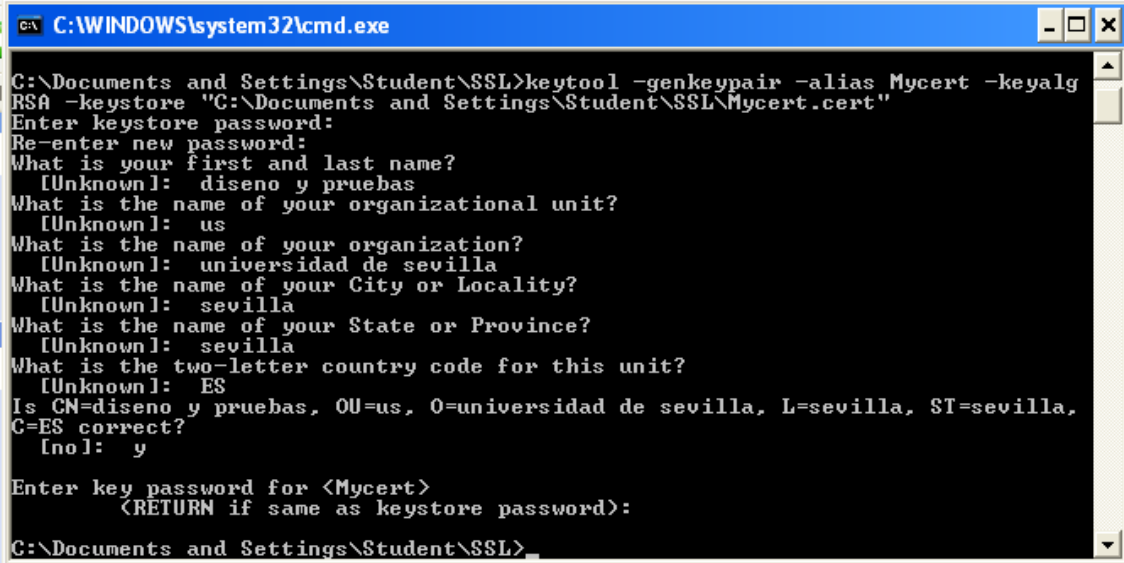
Pasos a seguir para configurar el servidor y el proyecto

Es importante que sigamos los siguientes pasos para conseguir que el protocolo HTTPS funcione correctamente en nuestra aplicación:

PASO 1: Debemos crear el certificado de seguridad. Para ello utilizaremos abriremos una CMD y escribiremos el siguiente comando:

Keytool -genkeypair -alias Mycert -keyalg RSA -keystore "ruta en la que se va a guardar el certificado".

Como vemos, se nos pedirá contestar a algunas preguntas, entre ellas la elección de una contraseña y algunos datos personales. Tan solo debemos contestar a las preguntas y ya tendremos el certificado Mycert.cert creado en la ruta escogida.



```
C:\WINDOWS\system32\cmd.exe

C:\Documents and Settings\Student\SSL>keytool -genkeypair -alias Mycert -keyalg
RSA -keystore "C:\Documents and Settings\Student\SSL\Mycert.cert"
Enter keystore password:
Re-enter new password:
What is your first and last name?
[Unknown]: disenyo y pruebas
What is the name of your organizational unit?
[Unknown]: us
What is the name of your organization?
[Unknown]: universidad de sevilla
What is the name of your City or Locality?
[Unknown]: sevilla
What is the name of your State or Province?
[Unknown]: sevilla
What is the two-letter country code for this unit?
[Unknown]: ES
Is CN=disenyo y pruebas, OU=us, O=universidad de sevilla, L=sevilla, ST=sevilla,
C=ES correct?
[no]: y
Enter key password for <Mycert>
(RETURN if same as keystore password):
C:\Documents and Settings\Student\SSL>
```

(Nota: en este caso la keystore password y la key password que estamos usando es "pass123").

PASO 2: Una vez creado el certificado, debemos configurar Tomcat para que acepte el protocolo HTTPS. Para hacerlo, debemos editar el archivo server.xml del servidor Tomcat. En concreto, debemos fijarnos en las líneas comentadas que comienzan por "Connector":

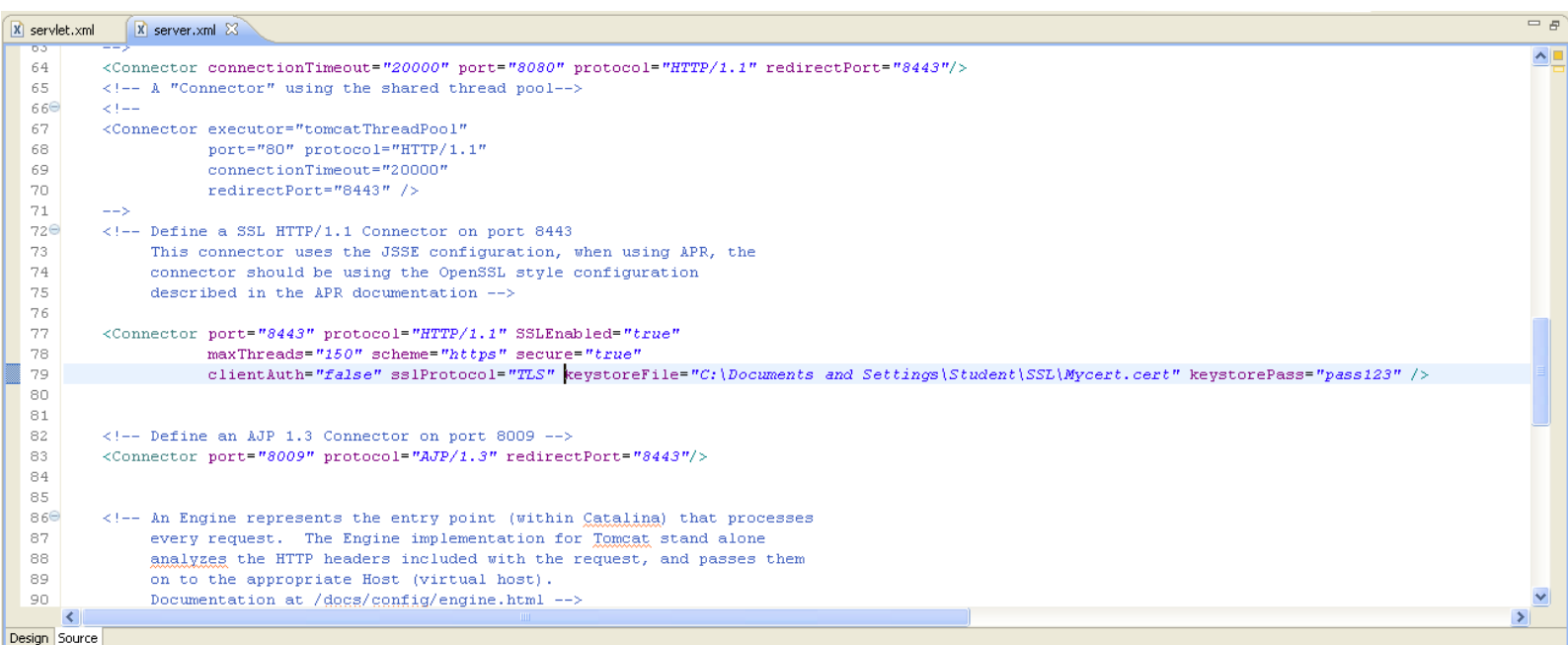


```
64 <Connector connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443"/>
65 <!-- A "Connector" using the shared thread pool-->
66 <!--
67 <Connector executor="tomcatThreadPool"
68         port="80" protocol="HTTP/1.1"
69         connectionTimeout="20000"
70         redirectPort="8443" />
71 -->
72 <!-- Define a SSL HTTP/1.1 Connector on port 8443
73      This connector uses the JSSE configuration, when using APR, the
74      connector should be using the OpenSSL style configuration
75      described in the APR documentation -->
76 <!--
77 <Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
78         maxThreads="150" scheme="https" secure="true"
79         clientAuth="false" sslProtocol="TLS" />
80 -->
81
82 <!-- Define an AJP 1.3 Connector on port 8009 -->
```

Para configurar HTTPS en el servidor tan solo debemos eliminar el comentario de estas líneas y añadir los siguientes atributos:

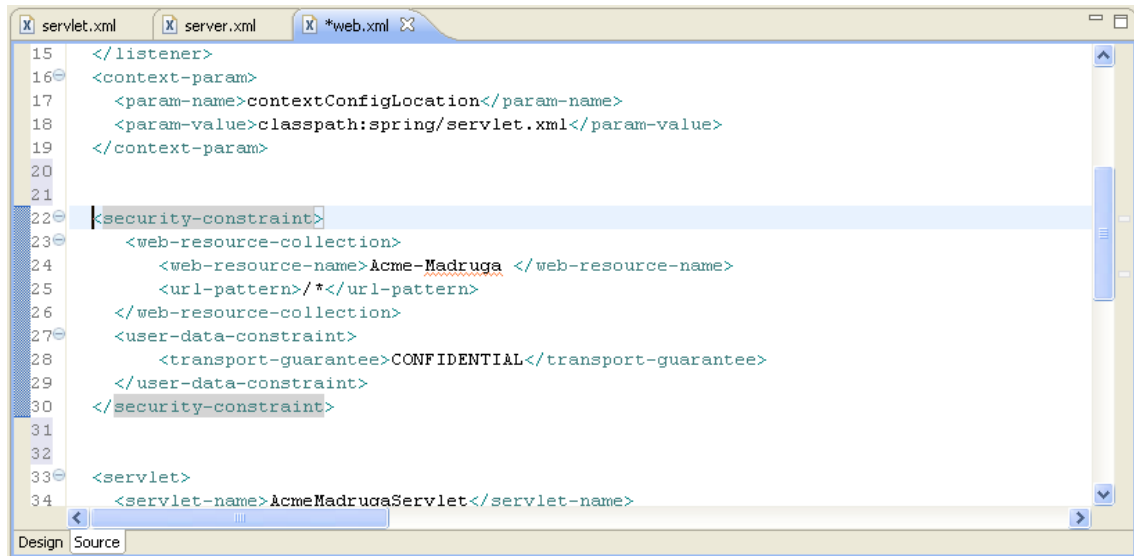
- KeystoreFile: la ruta de instalación indicada en el paso 1.
- KeystorePass: la contraseña indicada al final del paso 1 (en este caso, como se ve en la consola, usamos la misma contraseña que al principio).

En nuestro caso quedaría de la siguiente forma:



```
64 <Connector connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443"/>
65 <!-- A "Connector" using the shared thread pool-->
66 <!--
67 <Connector executor="tomcatThreadPool"
68         port="80" protocol="HTTP/1.1"
69         connectionTimeout="20000"
70         redirectPort="8443" />
71 -->
72 <!-- Define a SSL HTTP/1.1 Connector on port 8443
73      This connector uses the JSSE configuration, when using APR, the
74      connector should be using the OpenSSL style configuration
75      described in the APR documentation -->
76 <!--
77 <Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
78         maxThreads="150" scheme="https" secure="true"
79         clientAuth="false" sslProtocol="TLS" keyStoreFile="C:\Documents and Settings\Student\SSL\MyCert.cert" keyStorePass="pass123" />
80 -->
81
82 <!-- Define an AJP 1.3 Connector on port 8009 -->
83 <Connector port="8009" protocol="AJP/1.3" redirectPort="8443"/>
84
85
86 <!-- An Engine represents the entry point (within Catalina) that processes
87      every request. The Engine implementation for Tomcat stand alone
88      analyzes the HTTP headers included with the request, and passes them
89      on to the appropriate Host (virtual host).
90      Documentation at /docs/config/engine.html -->
```

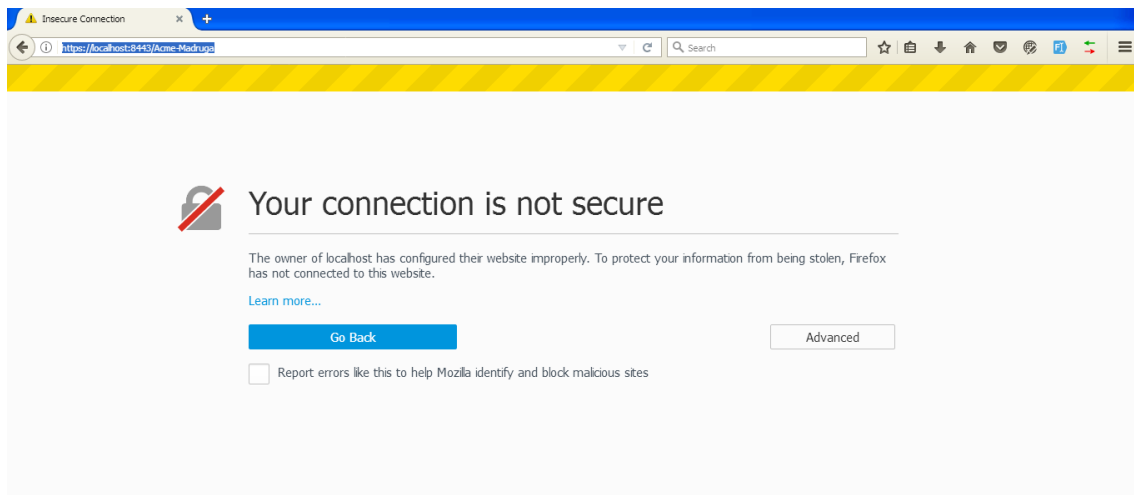
PASO 3: Debemos configurar nuestro proyecto para que admita el uso de HTTPS. Para ello debemos añadir el siguiente fragmento de código al archivo web.xml del proyecto:



```
15 </listener>
16 <context-param>
17   <param-name>contextConfigLocation</param-name>
18   <param-value>classpath:spring/servlet.xml</param-value>
19 </context-param>
20
21
22 <security-constraint>
23   <web-resource-collection>
24     <web-resource-name>Acme-Madruga </web-resource-name>
25     <url-pattern>/*</url-pattern>
26   </web-resource-collection>
27   <user-data-constraint>
28     <transport-guarantee>CONFIDENTIAL</transport-guarantee>
29   </user-data-constraint>
30 </security-constraint>
31
32
33 <servlet>
34   <servlet-name>AcmeMadrugaServlet</servlet-name>
```

(Nota: el atributo web-resource-name debe variar dependiendo del proyecto).

Una vez hecho todo esto, sólo debemos iniciar el servidor y probar a entrar a <https://localhost:8443/Acme-Madruga>, donde encontraremos lo siguiente:



Ahí debemos pulsar sobre “Advanced”, y clicar en “Add Exception” para añadir una excepción a nuestro navegador que nos permita acceder a la página en cuestión.

Una vez añadida se nos redireccionará a la página de inicio que hayamos configurado, y habremos terminado con la configuración de HTTPS en el proyecto.