Tomcat por el puerto 80

En este apartado vamos a hacer funcionar a nuestro Tomcat y sus correspondientes Host Virtuales por el puerto estándar de HTTP, es decir, el puerto 80.
En primer lugar y para que no interfiera, si hubiera algún servicio ocupando el puerto 80, tendríamos que pararlo; por ejemplo, *Apache*:

```
sudo service apache2 stop
```

El siguiente paso será editar el archivo de configuración por defecto de *Tomcat*, */etc/default/tomcat9*, modificando la última línea de la siguiente forma:

```
AUTHBIND=yes
```

A continuación debemos crear un archivo en el directorio */etc/authbind/byport/* llamado 80 y dejarlo vacío, esto permitirá asignar un puerto a un usuario. Lo hacemos de la siguiente forma:

```
sudo touch /etc/authbind/byport/80
```

Le damos permisos y se lo asignamos al usuario *tomcat9*:

```
sudo chmod 500 /etc/authbind/byport/80
sudo chown tomcat9 /etc/authbind/byport/80
```

Pasamos a editar el archivo */var/lib/tomcat9/conf/server.xml*, modificando el conector de la siguiente forma:

```
<Connector port="80" protocol="HTTP/1.1"
               connectionTimeout="20000"
               URIEncoding="UTF-8" />
```

El último paso consiste en reiniciar nuestro *Tomcat*, y ya lo tendremos funcionando por el puerto 80.

```
sudo service tomcat9 restart
```

Tomcat por el puerto 443

En este caso vamos a hacer funcionar a nuestro Tomcat por el puerto estándar HTTPS, es decir, el puerto 443. Este puerto aplica el protocolo de seguridad SSL sobre HTTP.
Los primeros pasos de esta práctica coinciden en gran parte con los de la configuración de Tomcat por el puerto 80 y deben realizarse únicamente si antes no se ejecutaron. Parar *Apache...*

```
sudo service apache2 stop
```

y editar el archivo de configuración por defecto de *Tomcat*, */etc/default/tomcat9*, modificando la última línea:

```
AUTHBIND=yes
```

Ahora debemos crear un archivo en el directorio */etc/authbind/byport/* esta vez llamado 443 y establecer los permisos y propietario de forma similar al ejercicio anterior:

```
sudo touch /etc/authbind/byport/443
sudo chmod 500 /etc/authbind/byport/443
sudo chown tomcat9 /etc/authbind/byport/443
```

A partir de este momento el proceso sigue un rumbo muy diferente, pues tendremos que establecer la seguridad mediante *SSL*, y para ello usaremos la herramienta keytool, que se gestiona los certificados para Java. El uso de esta herramienta va a resultar bastante más fácil que del *openssl*.

Necesitamos crear un almacén de claves y añadirle una clave

```
sudo keytool -genkey -alias miclave -keyalg RSA -keystore
/var/lib/tomcat9/mialmacen
```

En este punto nos irá pidiendo una serie de datos similares a cuando creamos la petición de certificado con openssl, pero en orden inverso. También nos solicitará las contraseñas para el almacén y la clave añadida.

Con todo esto podemos modificar el archivo *server.xml* y activar el puerto 443, dejando la directriz de la siguiente forma:

```
<Connector port="443" protocol="HTTP/1.1" SSLEnabled="true"
maxThreads="150" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
```

```
keystoreFile="/var/lib/tomcat9/mialmacen" keystorePass="123456"
keyAlias="miclave" keyPass="123456" />
```

A partir de ahora el gran problema es que si tenemos activados los puertos 80 y 443, se podrá acceder a los *Hosts Virtuales* independientemente del puerto, y la solución a este problema no la vamos a encontrar en *server.xml*, pues los puertos se configuran independientemente de los *Hosts*.

Esta vez la limitación se impondrá en las aplicaciones desplegadas y no en los *Hosts Virtuales*. Para ello tendremos que editar o en su caso crear el archivo web.xml en el directorio *WEB-INF* de la aplicación. Algo así:

```
gedit /var/lib/tomcat9/mihost/ROOT/WEB-INF/web.xml
```

Y tendremos que ponerle un contenido similar a este:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd">

<!-- redireccion a https -->
  <security-constraint>
     <web-resource-collection>
                <web-resource-name>Automatic SSL
Forward</web-resource-name>
                <url-pattern>/*</url-pattern>
     </web-resource-collection>
     <user-data-constraint>
                <transport-guarantee>
                     CONFIDENTIAL
                </transport-guarantee>
     </user-data-constraint>
  </security-constraint>

</web-app>
```

La marca *<transport-guarantee>* con su valor establecido en *CONFIDENTIAL* permitirá usar solamente esa aplicación por el puerto https.

Finalmente reiniciamos *Tomcat*.