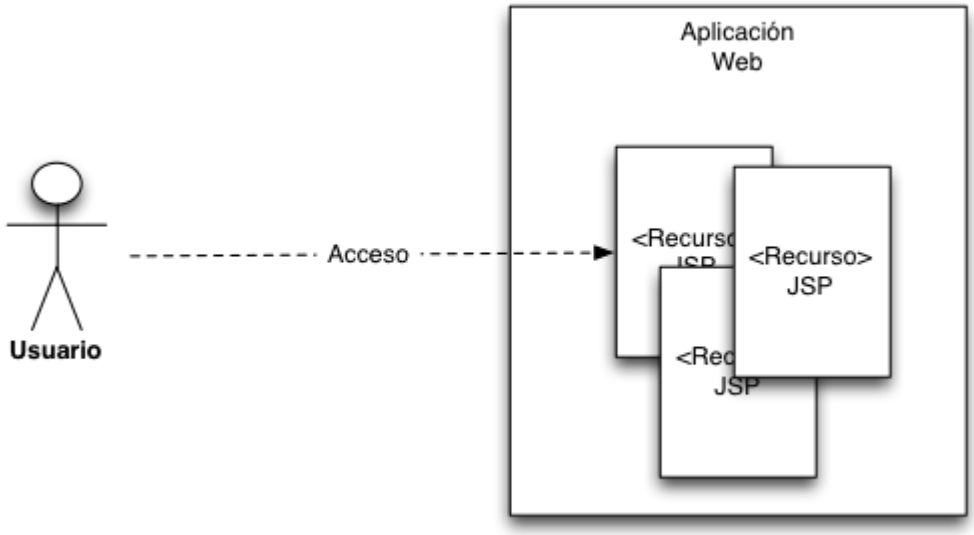
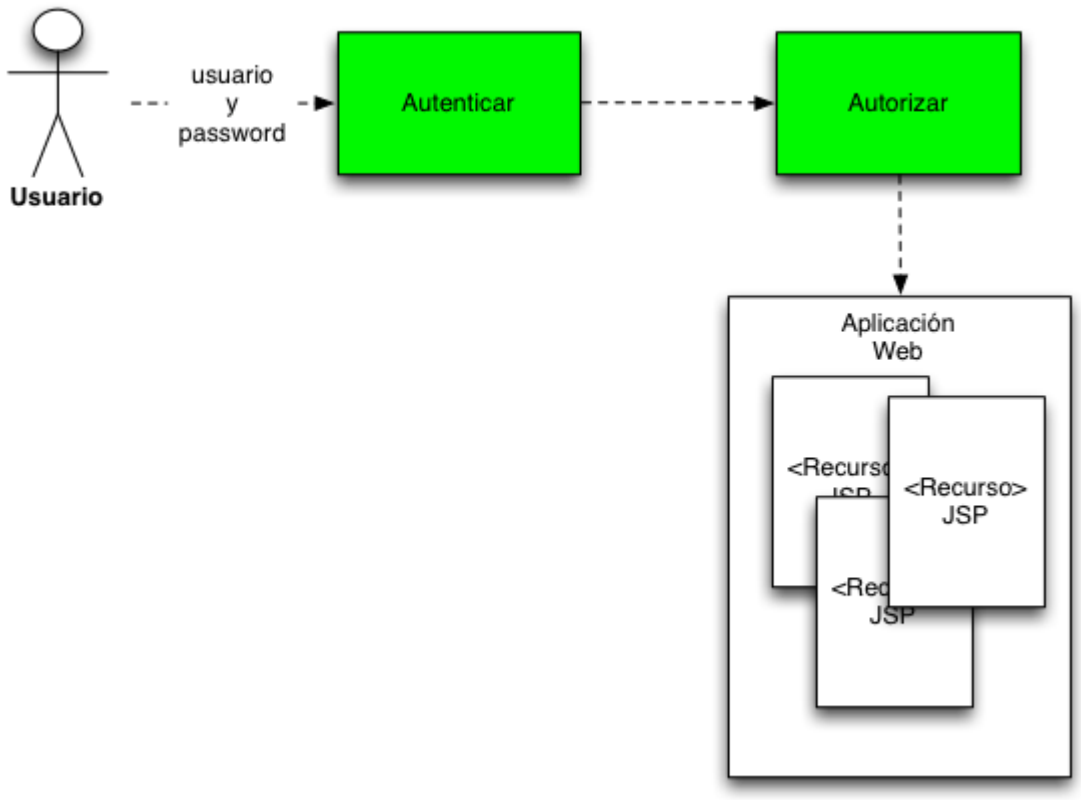
Uno de los temas que el libro no ha incluido es el tema de seguridad . Vamos en los próximos post ha realizar una pequeña introducción a como securizar una aplicación JEE usando el estándar de JAAS (Java Authentication and Authorization Service). Supongamos que un usuario desea controlar el acceso a unos recursos.

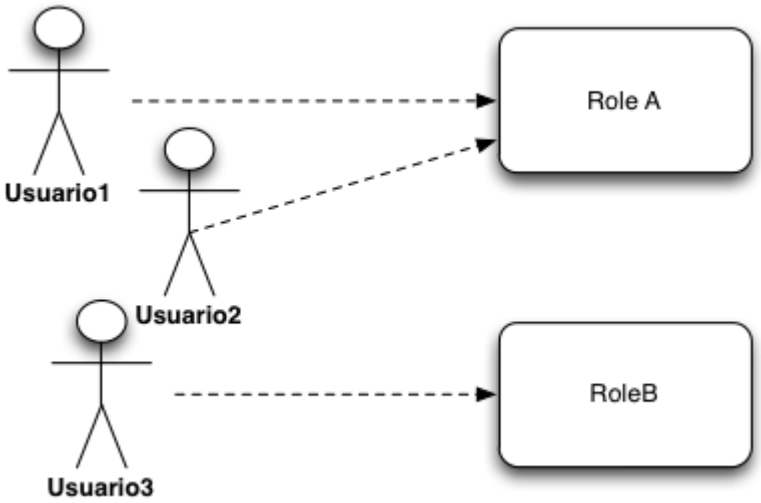


Normalmente ante este situación se pasa por dos fases la fase de Autenticación y la fase de Autorización.

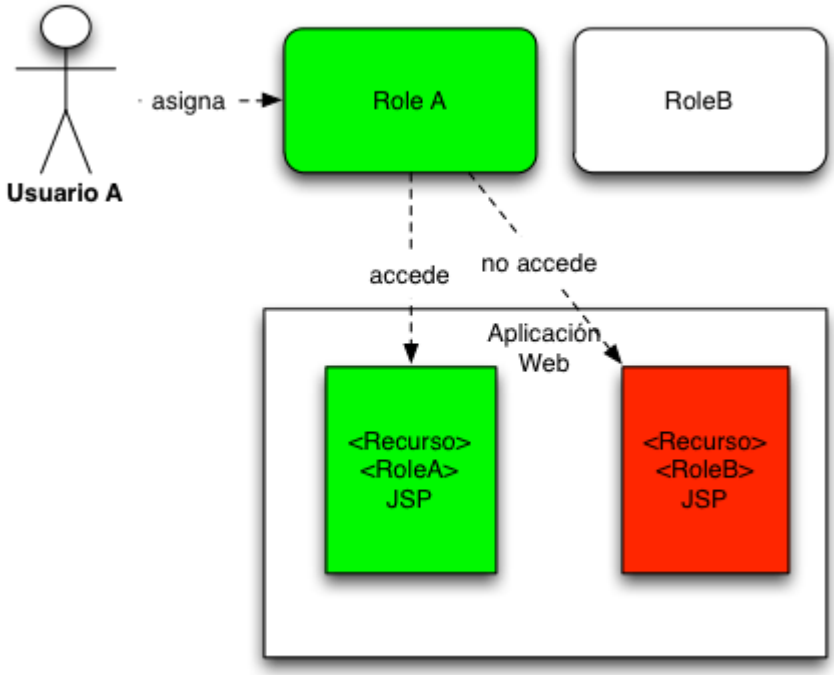


En la fase de Autenticación el usuario envia al servidor su nombre y su password y el servidor comprueba contra algún tipo de repositorio si el usuario es válido o no.Una vez que el usuario ha sido validado pasamos la segunda fase que es la fase de Autorización en la que se comprueba que el usuario tiene permisos para acceder a estos recursos.

Ahora bien en el standard de JAAS la fase de Autorización esta ligada a la gestión de roles . Es decir para que no tengamos que asignar permisos a los recursos para cada usuario individualmente se realiza un mapeo entre usuarios y roles .De tal forma que un conjunto de usuarios comparten un mismo ROL.



Una vez que tenemos claro el concepto de ROL . JAAS se encarga de definir cuales son los recursos a los que cada usuario a traves del ROL o ROLES que disponga puede acceder.



Vamos a ver a continuación como el standard JAAS se encarga de definir los recursos y roles necesarios para una aplicación web. Para ello deberemos modificar el fichero web.xml y añadirle etiquetas de seguridad.

```
<web-app xmlns=”http://java.sun.com/xml/ns/j2ee”
xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”
xsi:schemaLocation=”http://java.sun.com/xml/ns/j2ee web-app_2_4.xsd”
version=”2.4″>

<security-constraint>
<web-resource-collection>
<web-resource-name>recursosRoleA</web-resource-name>
<url-pattern>
/recursosRoleA/*
</url-pattern>
<http-method>GET</http-method>
<http-method>POST</http-method>
</web-resource-collection>
<auth-constraint>
<role-name>ROLEA</role-name>
</auth-constraint>
</security-constraint>

<security-constraint>
<web-resource-collection>
<web-resource-name>recursosRoleB</web-resource-name>
<url-pattern>
/recursosRoleB/*
</url-pattern>
<http-method>GET</http-method>
<http-method>POST</http-method>
```

```
</web-resource-collection>
<auth-constraint>
<role-name>ROLEB</role-name>
</auth-constraint>
</security-constraint>

</webapp>
```

Como podemos ver estamos definiendo dos restricciones de seguridad de acceso a recursos una denomina recursosRoleA y otra denominada recursosRoleB . La etiqueta de <url-pattern>define que recursos son protegidos y la etiqueta <role-name> que roles tienen acceso a ellos. Una vez realizada esta operación habremos terminado la fase de Autorización en el siguiente post cubriremos la parte de Autentificación.