**DAW Práctica 3.5: Mecanismos de seguridad en Apache Tomcat**

1. **Inicia nuestro servidor de aplicaciones de Apache y accede desde la URL a localhost:8080/manager. ¿Hay algún problema?**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Pide autentificación

1. **Dentro del directorio “conf” de Tomcat, ¿qué utilidad tiene el archivo tomcat-users.xml?**

Forma, Rectángulo

Descripción generada automáticamente

El archivo tomcat-users.xml tiene como utilidad la gestión de usuarios y roles en Apache Tomcat

1. **Siguiendo la guía que proporciona el propio fichero, crea un usuario con capacidad para acceder a la interfaz gráfica de manager (manager-gui). ¿Qué funciones tienen los diferentes roles que aparecen en el fichero?**

Texto

Descripción generada automáticamente

1. **Al acceder a manager nos muestra información interesante como las aplicaciones lanzadas o el estado del servidor (añade una captura de estos apartados) además de la posibilidad de gestionar las aplicaciones que tenemos iniciadas o desplegar alguna más añadiendo el .war entre otras funcionalidades.** Interfaz de usuario gráfica, Texto, Aplicación

   Descripción generada automáticamente

**Interfaz de usuario gráfica, Aplicación, Tabla

Descripción generada automáticamente**

1. **Implementaremos la necesidad de autentificación para una de nuestras aplicaciones. Para ello, modificaremos su archivo web.xml desde Eclipse añadiendo las siguientes instrucciones:**

**Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente**

**(1) definen el nombre del recurso al que le vamos a aplicar la configuración de seguridad (puede ser cualquier nombre) y el patrón de las URLs a las que va a aplicar. En este caso como hemos puesto /\* se aplicará a toda la aplicación.**

**(2) define el rol que deberán tener los usuarios que tengan permiso para conectarse.**

**(3) configura el tipo de autenticación (Basic). También se podría definir una base de datos externa en la que almacenar los usuarios para aumentar la seguridad.**

**Texto

Descripción generada automáticamente**

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto, Correo electrónico

Descripción generada automáticamente

1. **Una vez configurada la aplicación nos quedaría configurar los usuarios dentro del servidor. Para ello deberás modificar el archivo tomcat-users.xml añadiendo el siguiente contenido (el nombre del rol debe coincidir con el especificado en el archivo web.xml):**



1. **Solo quedaría actualizar el .war, reiniciar el servidor y probar que al acceder nos pide autenticación. Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

   Descripción generada automáticamente**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. **Ahora configuraremos el protocolo seguro HTTPS. Para ello, haremos uso de la herramienta Keytool. Keytool es una utilidad de línea de comandos incluida con el JDK (Java Development Kit) que se utiliza para administrar claves y certificados de seguridad.**
2. **Abriremos la consola en modo administrador y ejecutaremos el siguiente comando (define la utilidad de las extensiones que aparecen en el comando): keytool -genkey -alias tomcat -keyalg RSA -keystore keystore.jks -keysize 2048**

**Texto

Descripción generada automáticamente**

1. **Si no puedes ejecutar el comando, piensa que es una herramienta que proporciona el JDK.**

Añado las variables de entorno

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. **Responde a las preguntas de información para generar la clave y el certificado autofirmado.**

Contraseña: pruebas

Texto

Descripción generada automáticamente

1. **El archivo recientemente creado, que almacena tanto la clave como el certificado debe estar situado en RutaTomcat/conf. Podemos moverlo o crearlo directamente en esa dirección.**

Interfaz de usuario gráfica, Texto, Correo electrónico

Descripción generada automáticamente

1. **Deberemos editar el archivo server.xml que, si se han seguido los pasos anteriores, debería encontrarse en la misma carpeta que el archivo .jks creado. En dicho fichero deberemos modificar el conector de https o crearlo con la siguiente configuración en caso de que no esté implementado:**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. **Reinicia Apache tomcat y prueba a acceder a localhost desde el puerto implementado para HTTPS (**[**https://localhost:PUERTOIMPLEMENTADO**](https://localhost:PUERTOIMPLEMENTADO)**).**
2. **Es posible que no se tenga acceso si el puerto implementado no está abierto. Para abrirlo en Windows:**
3. **Ve al Panel de Control → Sistema y seguridad → Firewall de Windows Defender.**
4. **Selecciona Configuración avanzada.**
5. **Crea una nueva regla de entrada:** 
   1. **Tipo: Puerto.**
   2. **Protocolo: TCP.**
   3. **Puerto: 8443 (o el puerto configurado).**
   4. **Permitir la conexión.**
   5. **Asigna un nombre (por ejemplo: "Tomcat HTTPS").**

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. **Vuelve a intentar acceder utilizando HTTPS. El navegador nos muestra un aviso como hemos visto en prácticas anteriores (al ser un certificado autofirmado).**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. **Muestra los detalles del certificado implementado desde el navegador.**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

1. **Recuerda que dentro de Apache Tomcat tienes el directorio logs, donde podrás ver los diferentes motivos de posibles errores que te vayan sucediendo durante la implementación.**
2. **Utilizando el protocolo seguro intenta acceder a las aplicaciones creadas en prácticas anteriores.**

**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

1. **Investiga que componentes web tiene Apache Tomcat. ¿Se ha utilizado alguno durante la realización de las últimas prácticas?**

* Servlet Container: Gestiona y ejecuta los servlets, que son responsables de manejar las solicitudes HTTP y generar respuestas dinámicas.
* JavaServer Pages (JSP): Permite crear contenido web dinámico combinando HTML con código Java, convirtiendo las páginas JSP en servlets.
* Catalina: Es el contenedor principal de servlets de Tomcat, gestionando su ciclo de vida y las solicitudes HTTP.
* Coyote: El conector HTTP de Tomcat, responsable de manejar las conexiones de clientes (por ejemplo, navegadores), incluyendo el soporte para HTTPS.
* Manager App: Interfaz gráfica de administración que permite gestionar aplicaciones desplegadas, ver su estado y administrar el servidor.
* Host Manager: Permite gestionar múltiples hosts virtuales en un solo servidor Tomcat, útil para aplicaciones en diferentes dominios.
* JNDI: Proporciona acceso a recursos externos, como bases de datos o sistemas de mensajería, desde las aplicaciones web.
* Expression Language (EL): Simplifica el acceso a datos y facilita la manipulación de variables dentro de las páginas JSP.
* Security Realm: Gestiona la autenticación y autorización de usuarios, controlando el acceso a las aplicaciones.

Hemos utilizado Servelets