

**Tema**

DOMINIOS DE SEGURIDAD EN APLICACIONES JAVA ENTERPRISE EDITION

**Tutor**

Ing. Eduardo Mauricio Campaña Ortega

MIS. MDU.CCNA. CCIA.

PhD. (c) Ingeniería de Software

PhD. (c) Seguridad Información

**Fecha**

12/02/2022

DOMINIOS DE SEGURIDAD EN APLICACIONES JEE

[1. MARCO TEÓRICO 5](#_Toc119930256)

[1.1. INTRODUCCIÓN 5](#_Toc119930257)

[1.2. OBJETIVOS 5](#_Toc119930258)

[1.2.1. OBJETIVO GENERAL 5](#_Toc119930259)

[1.2.2. OBJETIVOS ESPECÍFICOS 5](#_Toc119930260)

[1.3. DOMINIOS DE SEGURIDAD EN APLICACIONES WEB 6](#_Toc119930261)

[1.3.1. DOMINIO PAYARA 6](#_Toc119930262)

[1.3.2. DESCRIPCIÓN DEL PROCESO DE IMPLEMENTACIÓN DE UN SISTEMA DE AUTENTIFICACIÓN BASADO EN FORMULARIO 7](#_Toc119930263)

[1.3.3. TERMINOLOGÍA DE SEGURIDAD EN APLICACIONES JAVA ENTERPRISE EDITION. 9](#_Toc119930264)

[1.3.4. DEPLOYMENT DESCRIPTOR 10](#_Toc119930265)

[1.3.5. FRAGMENTOS JSPF 10](#_Toc119930266)

[1.3.6. MÉTODO POST 11](#_Toc119930267)

[1.3.8. J\_SECURITY\_CHECK 12](#_Toc119930268)

[1.3.9. INCLUDE FILE = “” 12](#_Toc119930269)

[2. PARTE PRÁCTICA 12](#_Toc119930270)

[2.1. CREACION DEL PROYECTO 12](#_Toc119930271)

[2.2. CODIFICACIÓN DE LOS ARCHIVOS .JSP 14](#_Toc119930272)

[2.1.1. CODIFICACIÓN DEL ARCHIVO login.jsp 14](#_Toc119930273)

[2.1.2. CODIFICACION DEL ARCHIVO loginError.jsp 18](#_Toc119930274)

[2.3. CONFIGURAR DEPLOYMENT DESCRIPTOR web.xml 21](#_Toc119930275)

[2.4. CONFIGURAR PAYARA SERVER 23](#_Toc119930276)

[2.5. CODIFICACION DEL ARCHIVO admin.jsp 28](#_Toc119930277)

[2.6. INCLUSIÓN DE FRAGMENTOS JSP 32](#_Toc119930278)

[2.7. FUNCIONALIDAD 37](#_Toc119930279)

[2.6.1 ESTRUCTURA DE LA APLICACIÓN 37](#_Toc119930280)

[2.6.2 EJECUCIÓN DE LA APLICACIÓN 37](#_Toc119930281)

[3. CONCLUSIONES 39](#_Toc119930282)

[4. RECOMENDACIONES 39](#_Toc119930283)

[5. REFERENCIAS 39](#_Toc119930284)

**INDICE DE IMÁGENES**

[Figura 1. Creación Nuevo Proyecto 12](#_Toc119864000)

[Figura 2. Categorías para proyectos Java 12](#_Toc119864001)

[Figura 3. Nombre y ruta del nuevo proyecto 13](#_Toc119864002)

[Figura 4. Selección del servidor y version de Java EE 13](#_Toc119864003)

[Figura 5. Frameworks para el nuevo proyecto 14](#_Toc119864004)

[Figura 6. Creación de login JSP 14](#_Toc119864005)

[Figura 7. Definición de propiedades para el archivo login.jsp 14](#_Toc119864006)

[Figura 8. Codificación del archivo login.jsp 15](#_Toc119864007)

[Figura 9. Creación del login error JSP 15](#_Toc119864008)

[Figura 10. Definición de propiedades del archivo loginError.jsp 15](#_Toc119864009)

[Figura 11. Codificación para el archivo loginError.jsp 16](#_Toc119864010)

[Figura 12. Creación del archivo web.xml 16](#_Toc119864011)

[Figura 13. Categoría Web para web.xml 17](#_Toc119864012)

[Figura 14. Propiedades para el archivo web.xml 17](#_Toc119864013)

[Figura 15. Código generado en el archivo web.xml 18](#_Toc119864014)

[Figura 16. Configuración de seguridad en web.xml 19](#_Toc119864015)

[Figura 17. Creación del archivo payara-web.xml 19](#_Toc119864016)

[Figura 18. Categoría Payara para payara-web.xml 20](#_Toc119864017)

[Figura 19. Características para el archivo payara-web.xml 20](#_Toc119864018)

[Figura 20. Mensaje de error al crear el archivo payara-web.xml 21](#_Toc119864019)

[Figura 21. Configuración para el archivo payara-web.xml 21](#_Toc119864020)

[Figura 22. Encendido servidor Payara 21](#_Toc119864021)

[Figura 23. Abrir panel de administración Payara 22](#_Toc119864022)

[Figura 24. Panel de administración Payara 22](#_Toc119864023)

[Figura 25. Dominio file en Payara Server 23](#_Toc119864024)

[Figura 26. Tabla de usuarios en el dominio file 23](#_Toc119864025)

[Figura 27. Creación del usuario administrador 24](#_Toc119864026)

[Figura 28. Creación de la carpeta admin 24](#_Toc119864027)

[Figura 29. Propiedades para la carpeta file 25](#_Toc119864028)

[Figura 30. Creación de archivo admin en JSP 25](#_Toc119864029)

[Figura 31. Características para el archivo admin.jsp 26](#_Toc119864030)

[Figura 32. codificación para el archivo admin.jsp 26](#_Toc119864031)

[Figura 33. Creación de la carpeta jspf para fragmentos JSP 27](#_Toc119864032)

[Figura 34. Características de la carpeta jspf 27](#_Toc119864033)

[Figura 35. Creación del fragmento formularioLogin en JSP 28](#_Toc119864034)

[Figura 36. Propiedades para el archivo formularioLogin.jsp 28](#_Toc119864035)

[Figura 37. codificación para el archivo formularioLogin.jsp 29](#_Toc119864036)

[Figura 38. Codificación para el archivo login.jsp con el nuevo fragmento 29](#_Toc119864037)

[Figura 39. Codificación para el archivo errorLogin.jsp con el nuevo fragmento 30](#_Toc119864038)

[Figura 40. Ejecución de la aplicación web 30](#_Toc119864039)

[Figura 41. Estructura de la aplicación 30](#_Toc119864040)

[Figura 42. Página de inicio de sesión para usuarios admin 31](#_Toc119864041)

[Figura 43. Ingreso de datos en el formulario 31](#_Toc119864042)

[Figura 44. Página de bienvenida para usuarios admin 31](#_Toc119864043)

[Figura 45. Ingreso de un usuario no registrado en el formulario 32](#_Toc119864044)

[Figura 46. Página de error para usuarios admin no registrados 32](#_Toc119864045)

**ÍNDICE DE TABLAS**

[Tabla 1. Codificación del archivo index.js 14](file:///C:\Users\santi\Downloads\DOMINIOS%20SEGURIDAD%20MONSTER%20FINAL.docx#_Toc119870086)

[Tabla 2. Codificación del archivo loginError.js 17](file:///C:\Users\santi\Downloads\DOMINIOS%20SEGURIDAD%20MONSTER%20FINAL.docx#_Toc119870087)

[Tabla 3. Codificación del archivo admin.js 29](file:///C:\Users\santi\Downloads\DOMINIOS%20SEGURIDAD%20MONSTER%20FINAL.docx#_Toc119870088)

[Tabla 4. Codificación del archivo formularioLogin.js 33](file:///C:\Users\santi\Downloads\DOMINIOS%20SEGURIDAD%20MONSTER%20FINAL.docx#_Toc119870089)

# MARCO TEÓRICO

## INTRODUCCIÓN

Un dominio se define como un conjunto de propiedades, recursos e instancias de servidores de aplicaciones.

Un dominio de seguridad es una colección de activos que se encuentran protegidos generalmente bajo una persona administradora del sistema, además, un dominio permite diferenciar zonas de acceso de un sistema.

Conceptualmente un dominio incluye un conjunto de clases cuyas instancias tienen asignados los mismos permisos. Los dominios de protección se determinan mediante la política de seguridad activa en cada momento. El entorno Java mantiene una asociación entre el código (clases e instancias) y sus dominios de protección.

Los dominios de protección se determinan mediante la política de seguridad activa en cada momento. El entorno Java mantiene una asociación entre el código (clases e instancias) y sus dominios de protección [1].

Existen dos tipos de dominios de seguridad los cuales son:

1. ***Dominios de seguridad nativos:*** es un tipo de dominio que contiene usuarios y grupos creados los cuales son gestionados por medio de una herramienta Administrador, este dominio se crea de manera predeterminada en el momento de la instalación.
2. ***Dominios de seguridad de LDAP:*** este tipo de dominio contiene usuarios y grupos importados desde un servicio directorio de LDAP

## OBJETIVOS

### **OBJETIVO GENERAL**

Analizar y comprender la importancia de la configuración de dominios de seguridad en aplicaciones web y como realizar una conexión segura al servidor Payara, la cual es una implementación oficial de Java Enterprise Edition (JEE), donde se van a instalar certificados necesarios.

### **OBJETIVOS ESPECÍFICOS**

* Realizar un estudio del estado del arte sobre la seguridad en las aplicaciones web.
* Analizar la importancia de la implementación de los dominios de seguridad en las aplicaciones web.
* Analizar el proceso de autenticación de usuarios.
* Identificar y analizar el uso correcto de los roles de usuarios involucrados para el acceso a los recursos del sistema informático basado en las pollitas de seguridad establecidas.
* Comprobar estrategias sobre el almacenaje en base de datos de credenciales de usuario.
* Investigar y analizar los beneficios y posibilidades que nos proporciona Payara para lograr la implementación de los objetivos mencionados anteriormente.
* Desarrollar una aplicación implementando los conceptos mencionados anteriormente.
* Analizar el proceso autentificación de usuarios.
* Reconocer uso adecuado de los roles involucrados para acceder a los recursos en base a las políticas de seguridad establecida.

## DOMINIOS DE SEGURIDAD EN APLICACIONES WEB

En los últimos años el desarrollo de aplicaciones web se ha convertido en una de las industrias más evolutivas de la ingeniería del software.

Según Lujan Mora (2002), Las aplicaciones Web son aquellas herramientas donde los usuarios pueden acceder a un servidor Web a través de la red mediante un navegador determinado. Por lo tanto, se define como una aplicación que se accede mediante la Web por una red ya sea intranet o Internet. Por lo general se menciona aplicación Web a aquellos programas informáticos que son ejecutados a través del navegador.

Las aplicaciones web al permanecer en constante evolución y crecimiento es importante que sean implementadas manteniendo un concepto de los dominios de seguridad, ya que pueden ayudar a mantener una aplicación segura.

Por lo tanto, cuando se menciona un dominio de seguridad se hace referencia al análisis y creación de colecciones o grupos de usuarios con el objetivo de cumplir un rol en específico dentro de la aplicación web, aunque un usuario puede pertenecer a uno o varios grupos y con esto el servidor está en la capacidad de obtener toda la información acerca de la seguridad y de esta manera poder almacenarla ya sea en archivos planos, bases de datos o repositorios.

Cuando se habla de dominio de seguridad puede referirse a:

* Colecciones de Usuarios.
* Grupos de Seguridad.

Cada dominio de seguridad permite obtener información y guardarla en algún lugar de tal manera que esta se conserve.

Estos datos pueden ser almacenados en diversos lugares, por ejemplo:

* Archivos Planos.
* Base de Datos Relacional.
* Repositorios.

El objetivo del laboratorio es implementar una conexión segura al servidor Payara que permita realizar el intercambio de datos, donde el almacenamiento de las credenciales de los usuarios no sea vulnerado, y esto no presente un riesgo para la aplicación web.

### **DOMINIO PAYARA**

Payara es un servidor de aplicaciones de código abierto que es completamente compatible y fácil de utilizar.

Payara Server es un servidor de aplicaciones de código abierto derivado de GlassFish Server Open Source Edition. Fue creado en 2014 por C2B2 Consulting como reemplazo directo de GlassFish después de que Oracle anunciara que suspendería el soporte comercial para GlassFish. En comparación con GlassFish, Payara Server se lanza con mayor frecuencia trimestralmente con correcciones de errores, parches y mejoras adicionales. Desde abril de 2016, Payara Server está patrocinado por Payara Services Ltd, que es responsable del desarrollo continuo y de la coordinación de las contribuciones de la comunidad. Oficialmente compatible con Jakarta EE y MicroProfile («Payara Server», 2022).

Cada dominio de Payara Server tiene su propio archivo de política J2E global ubicado en la siguiente ubicación dir-dominio/config, el archivo se llama server.policy.

Esta sección cubre los siguientes temas:

* Permisos predeterminados
* Propiedades del sistema
* Cambio de permisos para una aplicación
* Habilitar y deshabilitar el administrador de seguridad [2].



**Ilustración 1. Logo Payara.**

### **DESCRIPCIÓN DEL PROCESO DE IMPLEMENTACIÓN DE UN SISTEMA DE AUTENTIFICACIÓN BASADO EN FORMULARIO**

Es un requerimiento muy común en cualquier aplicación web, el restringir el acceso a determinadas páginas solo para ciertos usuarios, en Java, se debe crear un dominio de seguridad en el servidor desde el cual se va a utilizar la aplicación. Los dominios de seguridad son colecciones de usuarios y grupos de seguridad, cada dominio de seguridad permite al servidor de la aplicación obtener información de seguridad de algún tipo y almacenarla de modo permanente.

Esta información de seguridad puede ser almacenada desde distintos dispositivos, es posible efectuar esto desde un archivo plano, a una base de datos relacional o a un repositorio. Pero configurar la aplicación para que se conserve la información segura de un almacenamiento persistente, libera el trabajo significativamente. Simplemente, se debe definir la autentificación de la aplicación en cuestión para que use un dominio de seguridad.

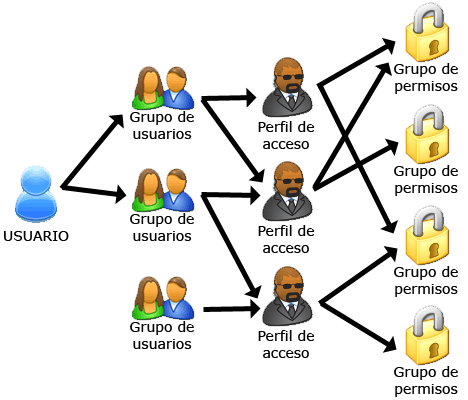


Ilustración 2. Dominios de seguridad

Cada usuario puede pertenecer a uno o más grupos de seguridad, las páginas seguras en una aplicación web son accesibles para ciertos grupos de seguridad, dicho procedimiento de configuración para los dominios de seguridad varía dependiendo del servidor que se esté utilizando, en concreto, se usará un dominio de seguridad conocido como file que Payara ya tiene configurado, este permite acceder a la consola de administración del servidor

Existen distintos modos para autentificar a los usuarios, pero el más común es la autentificación basada en formulario. Cuando se usa este tipo de autentificación, es necesario desarrollar una página HTML o JSP que es la que se usa para recoger las credenciales que indique el usuario; las ventajas de este método además de la habilidad para poder hacer páginas de LOGIN tan elaboradas como se desee usando HTML, CSS, JavaScript, etc., pero además de modo muy sencillo se puede encriptar la información configurando las páginas para que usen HTTP sobre SSL. Cuando usa este tipo de autenticación es necesario desarrollar una página HTML o JSP, la cual se utilizará para recolectar las credenciales que nos indique el usuario.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Ilustración 3. Proceso de Autentificación con Formulario.**

El proceso se inicia en el cliente, desde el cual se va a llevar a cabo la solicitud de un recurso protegido, solicitud que se hace al servidor Payara. Cuando el servidor detecta que se trata de un recurso protegido, redirecciona a la página de LOGIN (login.jsp), donde se incluye el formulario que tiene que completar el cliente, una vez completado el formulario es de nuevo enviado al servidor. Una vez que el formulario es enviado al servidor, es a j.security\_check, el mismo que es encargado de determinar si los datos que se han incluido son correctos o no. En caso de que los datos sean correctos, lo que hace es redireccionar al recurso que se había solicitado desde el principio. Por otro lado, si al comprobar los datos enviados en el login dan error, entonces se devuelve al cliente una segunda página de error indicando que el login es incorrecto y mostrando la posibilidad de que complete nuevamente el formulario.

Pasos para implementar un sistema de autentificación basado en formulario

1. Crear una página de login (inicio de sesión)
2. Crear una página de error (en caso de que las credenciales ingresadas sean incorrectas).
3. Configurar aplicación web en el servidor Payara.

Para el primer punto se utilizarán certificados generados por la herramienta FILE, ofrecida por Java. Con ella se crearán tres pares de claves y tres certificados, que equivaldrán a una Autoridad de Certificación, una Autoridad Intermedia y un usuario final. Estos se firmarán utilizando el algoritmo de hash SHA-256 para cumplir con lo exigido por los navegadores modernos. El certificado intermedio, el de usuario final y el par de claves de este deberán estar instalados en el servidor Payara, y el certificado raíz deberá ser importado como Entidad de Certificación Raíz de Confianza.

La gestión de la seguridad será realizada desde Security Real, que controlará el acceso de usuarios no identificados a zonas restringidas.

### **TERMINOLOGÍA DE SEGURIDAD EN APLICACIONES JAVA ENTERPRISE EDITION.**

La especificación de seguridad Java EE admite un conjunto de funcionalidades de seguridad requeridas, que incluyen autenticación, autorización, integridad de datos y seguridad de transporte para comprender de mejor manera como se define la seguridad en JEE se definen los siguientes conceptos:

Tabla 1. Terminología utilizada en Dominios de Seguridad en Java.

|  |  |
| --- | --- |
| TÉRMINO | DESCRIPCIÓN |
| Llamante o usuario: | Identidad con nombre individual definida en un almacén de identidad. |
| Credencial: | Contiene o hace referencia a información utilizada para autenticar a una persona que llama. |
| Grupo: | Permitir agrupar y clasificar un conjunto de usuarios que presentan características similares se pueden definir de distintas formas como interna (dentro de la aplicación) y externa (proveedor distinto a la autoridad pueden ser bases de datos).  Los usuarios de un grupo cuentan con permisos comunes. |
| Rol: | El objetivo de un rol al igual que de un grupo es el de clasificar a los usuarios, la diferencia que existe entre un grupo y un rol es que en un rol se especifican los permisos de acceso a los recursos de la aplicación, y finalmente los grupos pueden mapearse en uno o más roles.  Los roles son definidos dentro de la misma aplicación.  Se usa principalmente para especificar los permisos utilizados para proteger los recursos de la aplicación. La seguridad declarativa Java EE, como las restricciones en web.xml, solo funciona con roles, pero el uso de los recursos de la aplicación se puede proteger directamente a través de grupos si así se desea. |
| Entorno operativo: | Entorno en el que se implementa la aplicación Java EE y que, entre otros, contiene datos y sistemas que son externos a la aplicación Java EE. Para la seguridad de Java EE hay dos variaciones principales de este entorno.   * Entorno tradicional: los usuarios de seguridad y sus grupos se definen y administran externamente a la aplicación Java EE. * Entorno independiente: los usuarios de seguridad y sus grupos están definidos y gestionados internamente por la aplicación Java EE. |
| Mapeo de roles: | Proceso de mapear uno o más principales a uno o más roles. Un principal puede ser un grupo o un usuario / llamante, lo que da dos asignaciones semánticamente distintas:   * Mapeo de grupo a rol: proceso de mapear un grupo a un rol. En un entorno operativo tradicional con usuarios y grupos definidos externamente, este proceso se utiliza principalmente para asignar grupos externos a los roles locales definidos por la aplicación. * Mapeo de usuario / llamador a rol: procesamiento de mapear un usuario / llamador a un rol. Efectivamente, esto significa que un usuario / llamante conocido recibe directamente un rol de aplicación. Este tipo de mapeo se usa principalmente en un entorno operativo tradicional, ya que el nombre del usuario / persona que llama debe conocerse de antemano.   La especificación Java EE no aplica ni describe cómo se define ninguna de estas asignaciones o cómo se debe hacer la asignación. Tradicionalmente, esta asignación se define usando descriptores de implementación específicos del proveedor, por ejemplo, usando "ibm-application-bnd.xml" en Liberty o glassfish-web.xml en GlassFish y Payara, pero usando una combinación de JASPIC y JACC, esta asignación se puede hacer programáticamente de una manera específica de la aplicación también. |
| Mecanismo de autenticación: | Controlador que se encarga de la interacción del sistema de seguridad con la persona que llama y el entorno para verificar la identidad de la persona que llama. |

### **DEPLOYMENT DESCRIPTOR**

Describe las clases, recursos y configuración de la aplicación y la forma en que los usa el servidor web para entregar solicitudes web. Cuando el servidor web recibe una solicitud para la aplicación, usa el descriptor de implementación a fin de asignar la URL de la solicitud al código que debe manejarla.

### **FRAGMENTOS JSPF**

Para comprender que es un JSPF tenemos que definir que es un JSP el cual consiste en una tecnología que permite incluir código del lenguaje de programación Java en páginas web, donde el JavaServer Pages es un componente del servidor web el cual tiene el objetivo de obtener la página y sustituir el código Java que contiene el archivo y enviarla al cliente.

Un JSP contiene tres elementos principales los cuales pueden ser implementados en su desarrollo:

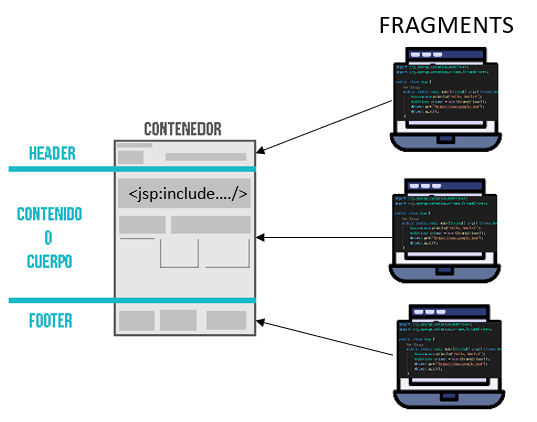
* Código: puede contener código Java incrustado la cual permite tener declaración de variables, métodos, expresiones y sentencias.
* Directivas: permite controlar distintos parámetros del servlet resultante de la traducción automática del JSP.
* Acciones: su funcionalidad es alterar el flujo normal de la ejecución de la página por ejemplo redirecciones, aunque puede tener otros usos.

Un archivo JSPF es utilizado en la programación de aplicaciones web con la finalidad de mantener un código más limpio y fácil de comprender, además, de reducir las líneas de código repetidos en nuestra aplicación para lo cual se realiza una división de la aplicación en varios unidades o módulos más pequeños y entendibles aplicando el concepto de divide y vencerás donde cada uno de los módulos tienen una única funcionalidad y puede ser reutilizable.

El archivo JSP contiene una directiva “INCLUDE” para cada uno de los archivos JSPF el cual se encarga de unir todas las partes para generar una aplicación.

En la mayoría de las páginas web y aplicaciones web se tiene una estructura de Cabecera (Header), Cuerpo (Body), Pie de página (Footer) y es en este punto la importancia de los Fragments en el desarrollo web.

El mantenimiento se facilita con estos fragmentos.



**Ilustración 4. Fragmentos JSP.**

Parte del código JSP pasado a un manejador de etiquetas que se puede invocar tantas veces como sea necesario. Puede pensar en un fragmento como una plantilla utilizada por un controlador de etiquetas para producir contenido personalizado. Por lo tanto, a diferencia de un atributo simple que es evaluado por el contenedor, un manejador de etiquetas evalúa un atributo de fragmento durante la invocación de etiqueta.

### **MÉTODO POST**

Es un método que consiste en el envío de información desde un cliente para la misma sea procesada en actualizar o agregar dicha información en el servidor, una solicitud POST no se almacena en el caché.

El método POST mantiene los datos ocultos del cliente y generalmente es utilizado en formularios.

**1.3.7.** **MÉTODO GET**

Es un método que nos permite obtener información del servidor ya sea de archivos planos, o de bases de datos y mostrarlos al cliente.

El método GET lleva los datos de forma visible para el cliente, la forma de envío es por medio de URL y todos los usuarios la pueden visualizar.

### **J\_SECURITY\_CHECK**

Se trata de autenticar el usuario y contraseña contra la base de datos Realm. Es un atributo action para los formularios de inicio de sesión, que forma parte de la API de Servlet.

Generalmente se encuentra incorporado en el directorio de archivos Tomcat. Es la técnica más utilizada para la validación de un login (usuario y contraseña) en las aplicaciones web o móviles, es por medio de un formulario de inicio de sesión, es el más común y recomendado por Java porque las cajas web puedan validar recursos de varias aplicaciones diferentes. El uso del atributo con las etiquetas HTML vendría siendo de esta forma, permitiendo especificar la acción correcta.

### **INCLUDE FILE = “”**

Inserta el contenido del fragmento JSP en la página, el valor del atributo FILE es la ruta relativa al archivo que se desea incluir.

# PARTE PRÁCTICA

## CREACION DEL PROYECTO

Crear un proyecto, clic en File y New Project.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figura . Creación Nuevo Proyecto

En la siguiente ventana, seleccionar la categoria Java Web, Web Application y clic en Next.

Graphical user interface, text

Description automatically generated

Figura . Categorías para proyectos Java

Definir un nombre para el proyecto, donde se guardara y clic en Next.

Text

Description automatically generated

Figura . Nombre y ruta del nuevo proyecto

Seleccionar como servidor de aplicaiones Payara Server, Java EE 7 Web y clic en Next.

Graphical user interface, text, application

Description automatically generated

Figura . Selección del servidor y version de Java EE

Para esta practica no se hara uso de ningun framework, clic en Next.

Graphical user interface, text, application

Description automatically generated

Figura . Frameworks para el nuevo proyecto

## CODIFICACIÓN DE LOS ARCHIVOS .JSP

### **CODIFICACIÓN DEL ARCHIVO login.jsp**

Sobre la carpeta Web Pages dar clic derecho, new y JSP.

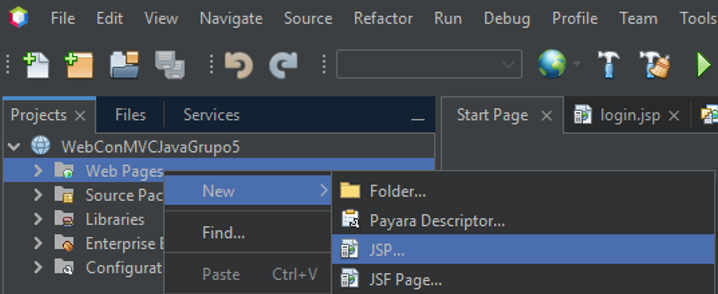


Figura . Creación de login JSP

En la siguiente ventana definir el nombre del archivo JSP como login y clic en Finish.

Graphical user interface, text, application

Description automatically generated

Figura . Definición de propiedades para el archivo login.jsp

En el archivo login.jsp borrar el código generado y colocar el siguiente código para construir la vista de inicio de sesión.

**Text

Description automatically generated**

Figura . Codificación del archivo login.jsp

Tabla . Codificación del archivo login.jsp

<%--

Document : login

Created on : Nov 19, 2022, 3:43:16 PM

Author : Grupo5

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Login</title>

</head>

<p> Por favor escribe tu nombre de usuario y contraseña para acceder a

la aplicación</p>

<form action="j\_security\_check" method="POST">

<table cellpadding="2" cellspacing="0" border="1">

<tr>

<td align="right">Nombre Usuario:&nbsp;</td>

<td>

<input type="text" name="j\_username">

</td>

</tr>

<tr>

<td align="right">Contraseña: &nbsp;</td>

<td>

<input type="password" name="j\_password">

</td>

</tr>

<tr>

<td></td>

<td> <input type="submit" value="Login" /></td>

</tr>

</table>

</form>

</html>

### **CODIFICACION DEL ARCHIVO loginError.jsp**

Sobre la carpeta Web Pages dar clic derecho, new y JSP.

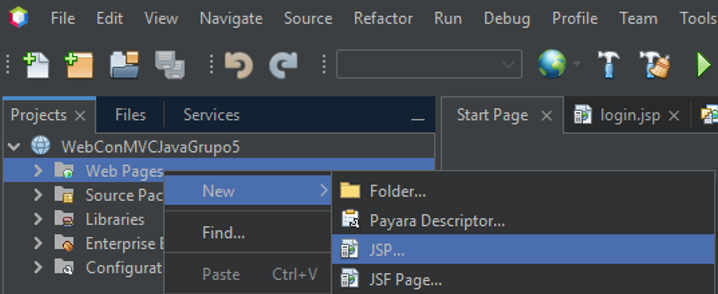
****

Figura . Creación del login error JSP

En la siguiente ventana definir el nombre del archivo JSP como loginError y clic en Finish.

**Text, application

Description automatically generated with medium confidence**

Figura . Definición de propiedades del archivo loginError.jsp

En el archivo loginError.jsp borrar el código generado y colocar el siguiente código para construir la vista de inicio error de sesión.

Text

Description automatically generated

Figura . Codificación para el archivo loginError.jsp

Tabla . Codificación del archivo loginError.js

<%--

Document : loginError

Created on : Nov 19, 2022, 3:53:57 PM

Author : Grupo5

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Login</title>

</head>

<body>

Se ha producido un error al hacer login. Inténtelo de nuevo, por favor.

<br />

<form action="j\_security\_check" method="POST">

<table cellpadding="2" cellspacing="0" border="1">

<tr>

<td align="right">Nombre Usuario:&nbsp;</td>

<td>

<input type="text" name="j\_username">

</td>

</tr>

<tr>

<td align="right">Contraseña: &nbsp;</td>

<td>

<input type="password" name="j\_password">

</td>

</tr>

<tr>

<td></td>

<td> <input type="submit" value="Login" /></td>

</tr>

</table>

</form>

</body>

</html>

## CONFIGURAR DEPLOYMENT DESCRIPTOR web.xml

Sobre la carpeta Web Pages dar clic derecho, New y Other.

Graphical user interface, text

Description automatically generated

Figura . Creación del archivo web.xml

Seleccionar la categoría Web, Standard Deployment Descriptor (web.xml) y clic en Next.

Graphical user interface, text, application

Description automatically generated

Figura . Categoría Web para web.xml

En la siguiente ventana dejar todos los campos por defecto y clic en Finish.

Text

Description automatically generated

Figura . Propiedades para el archivo web.xml

Se muestra un archivo con código XML y un panel de configuración.

Text

Description automatically generated

Figura . Código generado en el archivo web.xml

Dirigirse a la sección Security y rellenar los campos con la siguiente información:

* En la sección Login Configuration, indicar el tipo de autentificación form.
* En Form Login Page, escribir el nombre del archivo /index.jsp creado.
* En Form Error Page, escribir el nombre del archivo /loginerror.jsp creado.
* En Realm Name, indicar el nombre del dominio de seguridad que incluye Payara Server configurado que es file.
* En la sección Security Roles, añadir nuevos los roles de seguridad Admin y Usuario.
* En la sección Security Constraints, añadir un Web Resources con Name Páginas Administrativas y en URL Pattern(s), escribir /admin/\* la cual incluye en la restricción de seguridad a todas las páginas que se encuentren dentro de la subcarpeta llamada de dicha forma.
* Para hacer que todos los usuarios autorizados puedan ver las páginas protegidas, marcar la caja de chequeo Enable Authentification Constraint y en Role Name(s) seleccionar los nombres de los roles definidos para que puedan acceder a esta área de aplicación según se requiera, para este caso Admin.

Graphical user interface, application

Description automatically generated

Figura . Configuración de seguridad en web.xml

## CONFIGURAR PAYARA SERVER

En la carpeta Web Pages dar clic derecho, New y Other.

Graphical user interface, text

Description automatically generated

Figura . Creación del archivo payara-web.xml

Seleccionar la categoría Payara, Payara Descriptor y clic en Next.

A screenshot of a computer

Description automatically generated

Figura . Categoría Payara para payara-web.xml

En la siguiente ventana dejar todos los valores por defecto y clic en Next.

Graphical user interface, text, application

Description automatically generated

Figura . Características para el archivo payara-web.xml

A continuación, aparece un menaje de error, clic en OK y mover el archivo payara-web.xml a la carpeta WEB-INF.

Graphical user interface, text

Description automatically generated

Figura . Mensaje de error al crear el archivo payara-web.xml

En el archivo payara-web.xml dirigirse a la sección Security, clic sobre Add Security Role Mapping y rellenar los campos con la siguiente información:

* En Security Role Name definir Admin como rol.
* En group name dar clic sobre Add Group y definir appadmin como nombre de grupo.

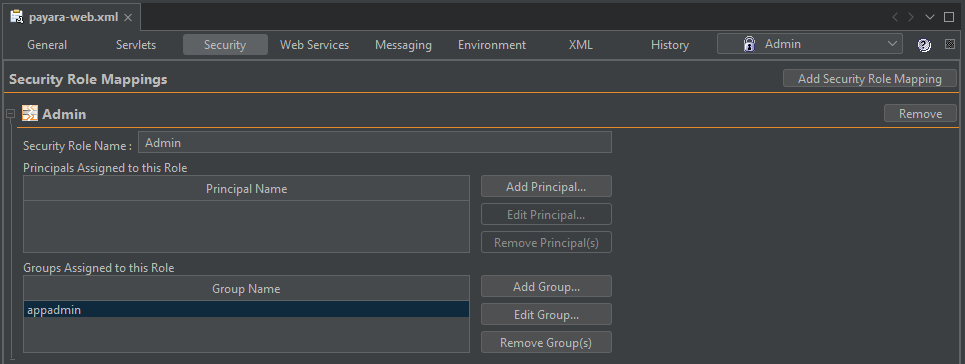


Figura . Configuración para el archivo payara-web.xml

Una vez finalizada la configuración del archivo payara-web.xml, dirigirse a Services, clic derecho sobre Payara Server y clic en Start.

Graphical user interface, application

Description automatically generated

Figura . Encendido servidor Payara

Cuando el servidor este iniciado, clic derecho sobre Payara Server y clic en View Domain Admin Console.

Graphical user interface, application

Description automatically generated

Figura . Abrir panel de administración Payara

A continuación, se abre la pestaña de configuración de Payara Server.

A screenshot of a computer

Description automatically generated

Figura . Panel de administración Payara

En la barra de navegación lateral, dirigirse a la sección Configurations, default-config, Security, Realms y clic en file.

A screenshot of a computer

Description automatically generated

Figura . Dominio file en Payara Server

En la tabla de usuarios, clic en New.

Graphical user interface

Description automatically generated

Figura . Tabla de usuarios en el dominio file

Crear un nuevo usuario que tendrá el rol Admin, para User ID ingresar un nombre de usuario, en Group List ingresar el grupo appadmin, definir una contraseña para el usuario y clic en OK para añadir el nuevo usuario.

Graphical user interface, website

Description automatically generated

Figura . Creación del usuario administrador

## CODIFICACION DEL ARCHIVO admin.jsp

Una vez creado el usuario, dirigirse a Apache NetBeans y dar clic derecho sobre la carpeta Web Pages, New y Folder.

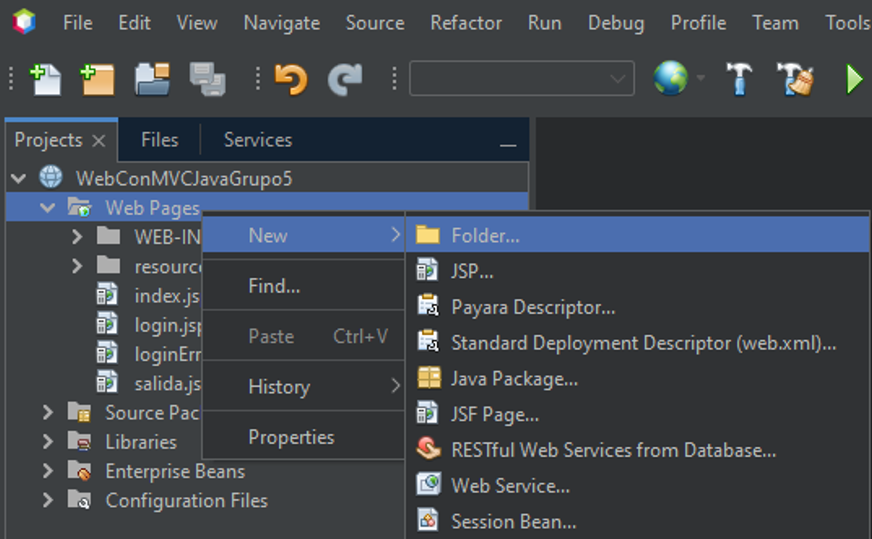


Figura . Creación de la carpeta admin

Para el Folder Name ingresar admin y clic en Finish.

Graphical user interface, text

Description automatically generated

Figura . Propiedades para la carpeta file

En la carpeta admin dar clic derecho, New y JSP.

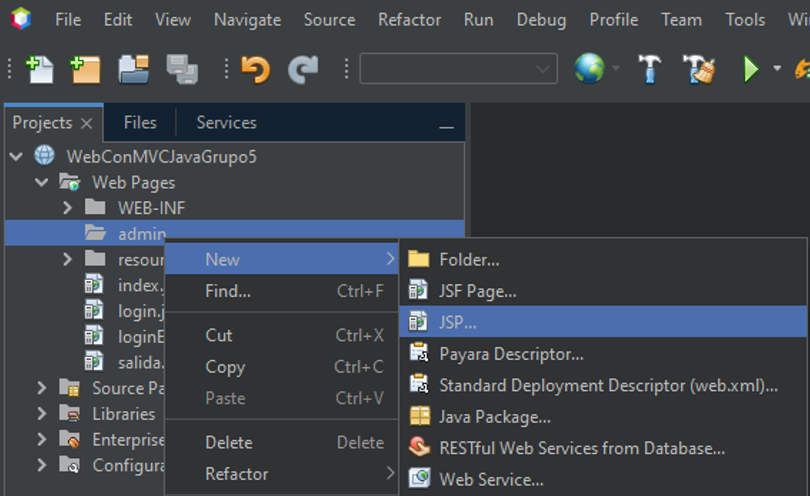


Figura . Creación de archivo admin en JSP

En la siguiente ventana definir el nombre del archivo como admin y clic en Finish.

Graphical user interface, text

Description automatically generated

Figura . Características para el archivo admin.jsp

En el archivo admin.jsp borrar el código generado y colocar el siguiente código para construir la vista de inicio del administrador.

Text

Description automatically generated

Figura . codificación para el archivo admin.jsp

Tabla . Codificación del archivo admin.js

<%--

Document : admin

Created on : Nov 19, 2022, 5:19:37 PM

Author : Grupo5

--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Página de administración</title>

</head>

<body>

<h1>Página de administración</h1>

<p>

Si esta fuera una página real de administración, podriamos administrar el sistema desde aquí.

</p>

</body>

</html>

## INCLUSIÓN DE FRAGMENTOS JSP

En la carpeta WEB-INF dar clic derecho, New y Folder.

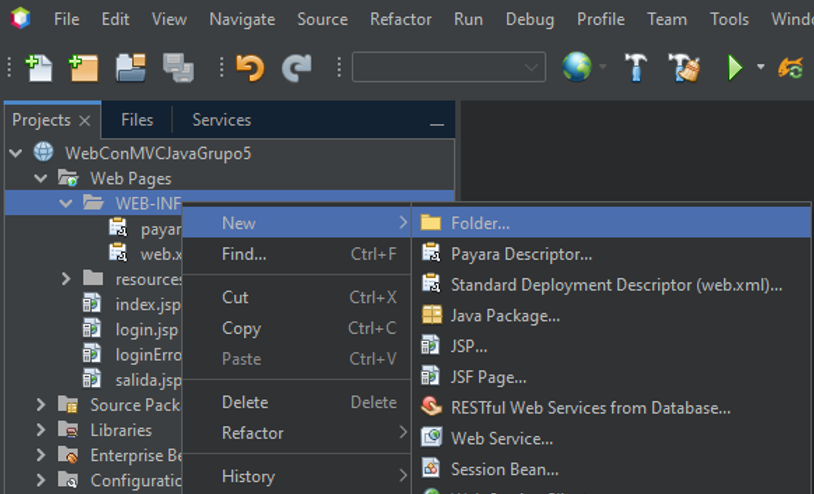


Figura . Creación de la carpeta jspf para fragmentos JSP

En la siguiente ventana definir el nombre de la carpeta como jspf y clic en Finish.

Graphical user interface, text

Description automatically generated

Figura . Características de la carpeta jspf

En la carpeta jspf dar clic derecho, New y JSP.

Graphical user interface, application

Description automatically generated

Figura . Creación del fragmento formularioLogin en JSP

En la siguiente ventana definir el nombre del archivo JSP como formularioLogin, marcar la opción Create as a JSP Segment y clic en Finish.

Text

Description automatically generated

Figura . Propiedades para el archivo formularioLogin.jsp

En el archivo formularioLogin.jsp borrar el código generado y colocar el siguiente código para definir un fragmento JSP que contendrá el formulario de las páginas de inicio de sesión (index.jsp) y fallo de inicio de sesión (erroLogin.jsp).

Text

Description automatically generated

Figura . codificación para el archivo formularioLogin.jsp

Tabla . Codificación del archivo formularioLogin.js

<%@ page pageEncoding="UTF-8" %>

<form action="j\_security\_check" method="POST">

<table cellpadding="2" cellspacing="0" border="1">

<tr>

<td align="right">Nombre Usuario:&nbsp;</td>

<td>

<input type="text" name="j\_username">

</td>

</tr>

<tr>

<td align="right">Contraseña: &nbsp;</td>

<td>

<input type="password" name="j\_password">

</td>

</tr>

<tr>

<td></td>

<td> <input type="submit" value="Login" /></td>

</tr>

</table>

</form>

Abrir el archivo login.jsp y remplazar el código del formulario por el JSP Fragment creado en el paso anterior.

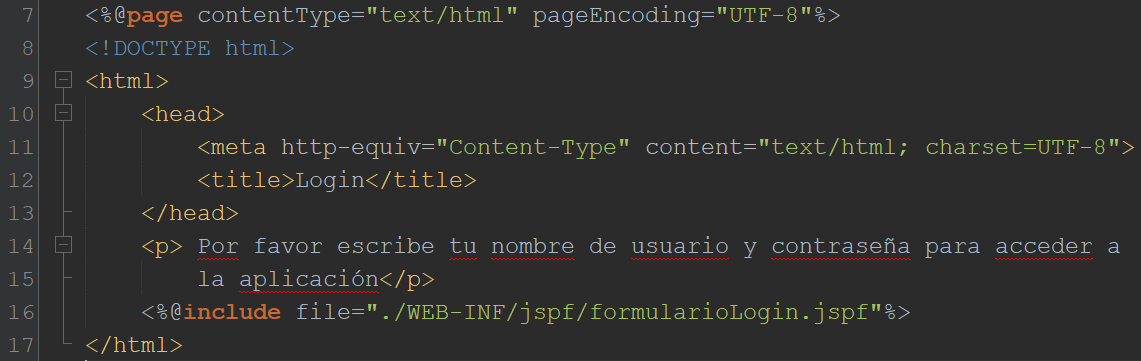


Figura . Codificación para el archivo login.jsp con el nuevo fragmento

Abrir el archivo erroLogin.jsp y remplazar el código del formulario por el JSP Fragment creado en el paso anterior.

Text

Description automatically generated

Figura . Codificación para el archivo errorLogin.jsp con el nuevo fragmento

Por último, guardar todos los cambios, clic derecho sobre el proyecto y clic en Run.

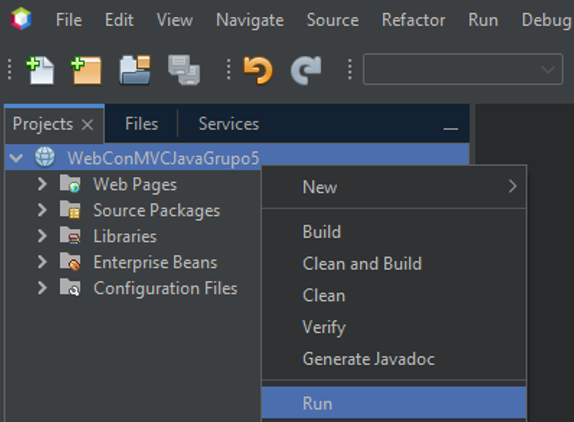


Figura . Ejecución de la aplicación web

## FUNCIONALIDAD

### **ESTRUCTURA DE LA APLICACIÓN**

Graphical user interface, text, application

Description automatically generated

Figura . Estructura de la aplicación

### **EJECUCIÓN DE LA APLICACIÓN**

#### **VISTA DE LA PAGINA PARA ADMINSITRADOR**

La carpeta a la que se le aplicaron restricciones de seguridad se llama admin y dentro de ella se ha colocado un archivo llamado admin.jsp, por lo tanto, para obtener acceso colocar: <http://localhost:8080/WebConMVCJavaGrupo5/admin/admin.jsp>

Se muestra la siguiente página:

Graphical user interface, application

Description automatically generated

Figura . Página de inicio de sesión para usuarios admin

Completar los campos requeridos de acuerdo con lo establecido anteriormente y presionar el botón de login.

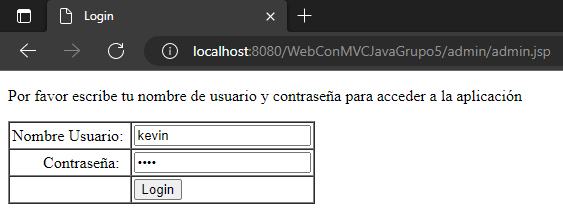


Figura . Ingreso de datos en el formulario

Si los datos son correctos, se presenta la siguiente página:

Graphical user interface, text, application

Description automatically generated

Figura . Página de bienvenida para usuarios admin

Si los datos son incorrectos se presentará la siguiente página de error de inicio de sesión.

Graphical user interface, application

Description automatically generated

Figura . Ingreso de un usuario no registrado en el formulario

Mientras no se ingrese un usuario y contraseña que este registrado se seguirá mostrando la página de erro de inicio de sesión.

Graphical user interface

Description automatically generated with medium confidence

Figura . Página de error para usuarios admin no registrados

# CONCLUSIONES

* Se concluye que, el dominio está bajo la responsabilidad de un administrador, es decir, una persona que define y aplica la política de seguridad para proteger el contenido dentro del dominio de seguridad que contiene un conjunto de información y redes bajo una política de seguridad común
* Se finiquita que, el dominio de seguridad creado interacciona con varias infoesferas mediante diferentes reglas, estas controlan los datos dentro del dominio de seguridad donde se encuentran.
* Se concluye que, este tipo de seguridad proporciona el aislamiento de las aplicaciones y los requisitos necesarios para autenticar a los usuarios de la aplicación.

# RECOMENDACIONES

* Se recomienda actualizar a la versión correspondiente cada cierto tiempo. Desde el propio Panel de Control podemos habilitar la opción para que se haga de manera automática mediante un mensaje de aviso.
* Se sugiere evitar ejecutar aplicaciones desconocidas, ya que lo mejor es no aceptar un applet con un certificado de origen incierto o que no haya sido emitido por una autoridad de certificación de confianza.(*Seguridad en JAVA: Seguridad en el entorno Java: Dominios protegidos, modelo de permisos y políticas de*, s. f.)

# REFERENCIAS

[1] Conde, J. (08 de Marzo de 2012). 05.- Curso de Java EE. Dominios de seguridad en aplicaciones Java EE. Obtenido de https://www.youtube.com/watch?v=RhnNUdct-o0

[2] Ortega, J. (2018). Seguridad en aplicaciones Web Java. Madrid: RA-MA Editorial.

[3] Otero, A. (15 de Enero de 2017). Tutorial básico de Java EE. Obtenido de https://www.lawebdelprogramador.com/pdf/1907-Tutorial-basico-de-Java-EE.html

[4] Pavón, J. (2012). Aplicaciones Web/Sistemas Web. Obtenido de https://www.fdi.ucm.es/profesor/jpavon/web/41-J2EE.pdf

[5] Payara Server. (2022). En Wikipedia. https://en.wikipedia.org/w/index.php?title=Payara\_Server&oldid=1117395661

[6] Seguridad en JAVA: Seguridad en el entorno Java: Dominios protegidos, modelo de permisos y políticas de. (s. f.). Recuperado 21 de noviembre de 2022, de https://www.uv.es/sto/cursos/seguridad.java/html/sjava-33.html