

SEGURIDAD EN APLICACIONES WEB

Curso 2018/19

Práctica 1: Autenticación y TLS/SSL.

IMPORTANTE: Para realizar esta práctica no utilizar una XAMPP 7.2.12, ni otra versión 7.2..., ya que da problemas con los certificados de cliente.

Al descomprimir el archivo que se encuentra en la plataforma Moodle, se obtienen tres carpetas: *practica1*, *certificados* y *base de datos*. Pretende ser una aplicación web con información sobre el Master de Ingeniería Web, a la que se irán añadiendo diferentes formas de autenticación y de autorización.

Los archivos que inicialmente contiene la carpeta *Practica1* son los siguientes:

- *masterWeb.php* es la página principal de la aplicación, con enlaces a las diferentes asignaturas.
- *seguridad.php* es la página correspondiente a la asignatura Seguridad en Aplicaciones Web, que es la única implementada.
- *login.php* realiza la autenticación web, pidiendo el login y el password.
- *logincert.php* para autenticación con certificado digital de cliente.
- *registro.php* para almacenar los datos de un nuevo usuario en la base de datos.
- *noAuth.php* mensaje que se visualiza cuando la autenticación no es correcta.
- *captcha.php* visualiza un captcha.
- *matricula.php* es un archivo que realiza el proceso de matrícula de un alumno en diferentes asignaturas.
- *asignaturas.php* informa de las asignaturas en las que el alumno se encuentra matriculado.
- *includes/abrirdb.php* realiza la apertura de la base de datos.
- *includes/autenticado.php* comprueba que el usuario esté autenticado.
- *admin/admin.html* menú para la administración de la aplicación web. No están implementadas las tareas de administración.

Durante la realización de la práctica se podrán modificar, cambiar de ubicación o añadir los archivos que se consideren oportunos.

ENTREGA DE LA PRÁCTICA: Se construirá un documento (pdf o Word) en el que se recogerán las explicaciones que se pidan en este enunciado a través de la etiqueta documento. Además, en la carpeta *practica1* se entregará la aplicación resultante una vez finalizado el último apartado. Finalmente, en la carpeta *certificados* se entregarán los archivos relacionados con certificados digitales que se piden en este enunciado. Todo ello se comprimirá en un archivo zip o rar y se entregará a través de la plataforma Moodle. No se entregará la base de datos resultante.

Servidor virtual

Para hacer este apartado se requiere que el usuario tenga permiso de administrador en el sistema operativo. De no ser así, porque se vaya a realizar la práctica en los ordenadores de la sala del máster, se leerá el enunciado de este apartado y al final del mismo se indica la alternativa que se deberá realizar.

La idea de este apartado es definir dos sitios web virtuales en nuestro servidor web, ambos en la dirección local 127.0.0.1. Uno de ellos, www.miw.com, se utilizará para la aplicación web objeto de esta práctica. Mientras que el otro, localhost, se podrá utilizar para el resto de contenidos de otras asignaturas, *phpMyAdmin*, etc.

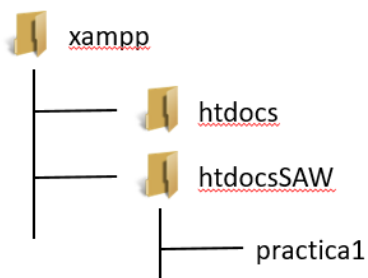
Para realizar este apartado es necesario que el usuario tenga permiso para modificar el fichero **hosts**, que en Windows se encuentra en C:\Windows\System32\drivers\etc\hosts.

A falta de un servidor DNS, se abrirá un editor en modo administrador y se editará el archivo **hosts**, añadiéndole las siguientes líneas que asocian los dominios localhost y www.miw.com a la dirección IP 127.0.0.1:

```
127.0.0.1    localhost
127.0.0.1    www.miw.com
```

Con ello, lo que se consigue es que nuestros navegadores web soliciten la IP 127.0.0.1 cuando se utilice como nombre de dominio localhost o www.miw.com.

Para localhost se almacenarán los contenidos en *htdocs*, mientras que para www.miw.com, se creará un nuevo directorio llamado *htdocsSAW*, al mismo nivel que *htdocs*. Dentro de *htdocsSAW* se colocará la carpeta *practica1* con los ficheros web de esta práctica.



En el fichero *xampp/apache/conf/extras/httpd-vhosts.conf* (incluido desde *httpd.conf*) se definirán los dos *Virtualhost*, con las siguientes consideraciones:

1. Para www.miw.com:
 - El *document root* será la carpeta *htdocsSAW*.
 - En el directorio del *document root*:
 - Estará permitido el acceso a todos los contenidos.
 - Se podrá cambiar la configuración a través de ficheros *.htaccess*.
 - En el directorio *practica1*:
 - No se podrá obtener listados de directorios (*directory browsing*).
 - En caso de realizar una petición a un directorio sin especificar recurso, el servidor Apache buscará *login.php*.

2. Para localhost, salvo el *ServerName*, no se definirá ninguna propiedad adicional, ya que hereda las directivas del servidor principal. Sin embargo, debe definirse este *VirtualHost* para evitar funcionamientos inadecuados en algunos casos.

Una vez definidos los dos *VirtualHost*, vuelva a arrancar Apache y realice peticiones a <http://www.miw.com/practica1> para comprobar el correcto funcionamiento sobre esta práctica. Igualmente, navegue por <http://localhost> para comprobar que sigue teniendo acceso al resto de contenidos.

Documento: explique todas las modificaciones realizadas al fichero *httpd-vhosts.conf* para definir los *VirtualHost*. No incluir el fichero *httpd-vhosts.conf* en la entrega.

Alternativa a servidor virtual si no se tiene permiso de administrador:

Si ya se ha realizado el apartado de servidor virtual, este apartado se saltará.

Al no tener permiso de administrador, el único dominio posible es localhost. Por ello, se utilizará un puerto diferente para definir el nuevo servidor virtual. Concretamente, en lugar de www.miw.com se utilizará **localhost:8090**.

En el fichero *xampp/apache/conf/extras/httpd-vhosts.conf* se incluirá la directiva *Listen 8090* y se definirán los dos servidores virtuales. Uno para localhost:8090, con las características indicadas para www.miw.com y otro para localhost.

Una vez definidos los dos *VirtualHost*, vuelva a arrancar Apache y realice peticiones a <http://localhost:8090/practica1> para comprobar el correcto funcionamiento sobre esta práctica. Igualmente, navegue por <http://localhost> para comprobar que sigue teniendo acceso al resto de contenidos.

Documento: explique todas las modificaciones realizadas al fichero *httpd-vhosts.conf* para definir los *VirtualHost*. No incluir el fichero *httpd-vhosts.conf* en la entrega.

Autenticación http básica.

2. En este apartado se definirá autenticación http básica para acceder a la parte de administración (carpeta *admin*). Para ello, se creará un fichero de usuarios con la aplicación **htpasswd**, incluyendo al menos un usuario administrador.

En el fichero *.htaccess* de la carpeta *admin*, se definirá la autenticación http básica basada en el fichero de usuarios creado anteriormente. Intenta acceder a *admin.html* para comprobar el correcto funcionamiento de la autenticación básica.

Por otra parte, en el fichero *.htaccess* del directorio *includes* se definirá que no es posible acceder por http a ninguno de sus recursos. Por ejemplo, si se pide <http://www.miw.com/includes/abrirbd.php>, se obtendrá un error 403 de acceso prohibido.

Documento: mostrar el contenido de los ficheros *.htaccess*.

Autenticación y autorización web.

La autenticación web de los apartados siguientes utiliza una base de datos, que contiene una única tabla *usuarios* con los siguientes campos: *id*, *user*, *password*, *salt*, *nombre*, *apellidos* y *permisos*. El primero es una clave numérica con autoincremento, y el resto son campos de tipo VARCHAR. Para importar la base de datos, se creará en MariaDB una base de datos llamada *datosmiw*, por ejemplo con *phpmyAdmin*. A continuación, se seleccionará dicha base de datos y se importará el fichero *datosmiw.sql*, contenido en la carpeta *Base de datos*.

Para que la aplicación realice el proceso de autenticación deben llevarse a cabo ciertas tareas:

3. Complete *login.php* de manera que compruebe si el *password* introducido por el usuario es el correcto. Para ello, habrá de tenerse en cuenta el *password* y el *salt* que ese usuario tiene en la base de datos. Se recomienda consultar en *registro.php* la forma en que se ha almacenado dicho *password* en la base de datos.
4. Defina el archivo *includes/autenticado.php* de manera que compruebe si el usuario está autenticado. En caso de no estarlo, redirigirá a *NoAuth.php* y finalizará con una instrucción *exit*. Por el contrario, si el usuario está autenticado, entonces no hará nada (permitiendo acceder a los contenidos).
5. Finalmente, las páginas que representan contenidos web (*seguridad.php*, *masterWeb.php*, *asignaturas.php* y *matricula.php*), utilizarán el archivo *autenticado.php* para asegurar que sólo son accesibles por usuarios autenticados.

Se pretenden ahora incorporar ahora comprobaciones relativas a la autorización. Para ello se utiliza el campo *permisos* de la tabla *usuarios* en la base de datos. Este campo de tipo VARCHAR está formado por 10 caracteres, cada uno asociado con una asignatura del máster. Por ejemplo, si el primer carácter es 'S', el usuario está autorizado a acceder a la información de la primera asignatura del Máster (Ingeniería Web: Visión General).

6. Añadir a la aplicación el código *php* necesario para que sólo tengan acceso a la información de *Seguridad en Aplicaciones Web* los usuarios que tengan permiso para ello. Esta asignatura se cursa es la sexta en cursas en el máster.

Nota: fijarse en *login.php* cómo se almacena en `$_SESSION['permisos']` los permisos del usuario autenticado en forma de *array* con la función *str_split*.

Documento: explica brevemente cómo lo has conseguido.

7. En este apartado se pretende introducir un *captcha* en *login.php*. Para lo cual se utilizará el fichero *captcha.php*. Se recomienda echar un vistazo a dicho fuente para ver cómo genera una cadena aleatoria de caracteres y la introduce en `$_SESSION['CAPTCHA']`, generando finalmente una imagen con dicha cadena.

Para incorporar el *captcha* en *login.php* se seguirán los siguientes pasos:

- Dentro del formulario de *login.php* se incluirá:

```
<img src= captcha.php>
<input type= text name= 'valor'>
```



Usuario:	<input type="text"/>
Password:	<input type="password"/>

	<input type="text"/>	LOGIN
---	----------------------	-------

REGISTRAR USUARIO

Nota: si no se visualiza la imagen del *captcha*, descomente en el fichero *php.ini* la línea: *extension=php_gd2.dll*


- Al comprobar en *login.php* los datos introducidos en el formulario, el campo 'valor' deberá contener la misma cadena que *captcha.php* almacenó en \$_SESSION['CAPTCHA']. En caso de no coincidir se redireccionará a *login.php*, visualizando además un mensaje del error. Si el *captcha* es correcto realizará el proceso de comprobar en la base de datos el *login* y el *password*.

De forma similar, añada un captcha en la página de registro para evitar que un robot pueda registrar usuarios de forma indefinida.



REGISTRO DE UN NUEVO USUARIO

Usuario:	<input type="text"/>
Password:	<input type="password"/>
Nombre:	<input type="text"/>
Apellidos:	<input type="text"/>

	<input type="text"/>	REGISTRO
---	----------------------	----------

[Volver a login](#)

Configuración SSL.

8. Crear un certificado digital para un servidor web asociado al dominio www.miw.com, firmado por la autoridad de certificación *Master Ingeniería Web*. Los archivos de dicha autoridad están incluidos en la carpeta *certificados* de la práctica: CAMIW.crt (certificado) y CAMIW.key (clave privada).

- Tareas asociadas al solicitante del certificado de servidor web: crear la pareja de claves (miw.key) y crear el *Certificate Signing Request* (miw.csr)
- Tareas asociadas a la autoridad de certificación *Master Ingeniería Web*: una vez recibido el CSR, lo firma para obtener el certificado digital de servidor web (miw.crt). El Common Name (CN) del certificado obtenido debe ser el dominio web, es decir, www.miw.com o localhost, en función de cómo se haya hecho el primer apartado de servidor virtual.

Nota: El *pass phrase* de la autoridad es “mica”.

Al entregar la práctica se incluirán los tres archivos mencionados (miw.key, miw.crs y miw.crt) en la carpeta *certificados*.

9. Configurar un *VirtualHost* para cada dominio SSL: <https://www.miw.com> y <https://localhost>.

Para definir los *VirtualHost* SSL, habrá de accederse al fichero *httpd-ssl.conf*. El *VirtualHost* definido por defecto se destinará para el dominio localhost, y únicamente habrá de cambiarse el *ServerName*. Por otra parte, se creará un nuevo *VirtualHost* para www.miw.com por el puerto 443, en el que se contemplarán todas las directivas del apartado 1 (el *document root* es *htdocsSAW*, etc). Además, se deberán incluir en este último *VirtualHost* las directivas necesarias para SSL, utilizando el certificado y la clave privada obtenidos en el apartado anterior. Ver transparencia 30 del tema 4.

Alternativa de este apartado para quien no tenga permiso de administrador: el servidor virtual SSL se definirá para <https://localhost:8443>, en lugar de para <https://www.miw.com>. En primer lugar, *httpd-ssl.conf* se incluirá la directiva *Listen 8443*, y a continuación se definirán los *VirtualHost* como se ha indicado en el párrafo anterior, excepto que en lugar de considerar el dominio www.miw.com se considerará el dominio **localhost:8443**.

Una vez definidos los *VirtualHost*, se intentará volver a arrancar el servidor Apache. Si se obtiene un error, se comprobará el correcto uso de las directivas y que los ficheros de certificado y clave privada se encuentran colocados en el sitio correcto.

Recuerde instalar la autoridad de certificación *Master de Ingeniería Web* (CAMIW.crt) como autoridad de confianza en el navegador cliente. No olvide marcar que es una autoridad de confianza para todo.

Compruebe el correcto funcionamiento de ambos dominios por HTTPS.

Documento: incluya la definición completa del *VirtualHost* correspondiente a <https://www.miw.com> (o a <https://localhost:8443>).

Utilizando la directiva *SSLRequireSSL*, configure ahora que la aplicación sólo admita peticiones *https*, no *http*.

Documento: Explica cómo has utilizado la directiva SSLRequireSSL. ¿Qué diferencia habría si en lugar de utilizar SSLRequireSSL se eliminara la directiva Listen 80 del fichero httpd.conf?

Coloque ahora la herramienta *Burp Suite* como *proxy* y navegue por la aplicación web por HTTPS.

Documento: ¿Por qué ha tenido que aceptar una excepción de seguridad para permitir el certificado? ¿Qué certificado digital está utilizando ahora la aplicación web? ¿Por qué motivo se está utilizando este certificado?

10. Modifique el archivo de configuración *httpd-ssl.conf* para que el protocolo SSL/TLS utilizado con un cliente no sea inferior a TLS 1.2.

Modifique el archivo de configuración *httpd-ssl.conf* para que la suite de cifrado que se negocie con el cliente cumpla las siguientes restricciones:

- Preferentemente se utilice el algoritmo de intercambio de clave ECDHE.
- Preferentemente se utilice el algoritmo de cifrado AES.
- Sólo se permitan algoritmos de cifrado con clave de al menos 256 bits (HIGH).
- No se permita utilizar el algoritmo RSA para intercambio de clave (kRSA).
- No se permita el algoritmo de cifrado RC4.
- No se permita MD5 ni SHA1 para hash.

Documento: Explique qué directivas ha definido en este apartado y cómo. ¿Qué algoritmos se obtienen cuando se realiza una petición a <https://www.miw.com> con el navegador? Añada un pantallazo de la ventana del navegador donde aparecen dichos algoritmos.

Autenticación de clientes con SSL.

Se supone que, una vez registrado un cliente en la aplicación, puede solicitar un certificado digital a la autoridad de certificación del *Master de Ingeniería Web*. El *Common Name* (CN) de dicho certificado será el *Usuario* que el cliente haya introducido en el proceso de registro. A partir de ese momento, el certificado podrá ser utilizado por el cliente para autenticarse en la aplicación web.

En los apartados siguientes se realizará paso a paso la creación del certificado personal, así como la instalación y configuración para poder utilizarlo.

11. Crear un certificado digital de persona física para el alumno que realiza la práctica:

- Tareas asociadas al solicitante del certificado: crear la pareja de claves (cliente.key), crear el Certificate Signing Request (cliente.csr)
- Tareas asociadas a la CA Master Ingeniería Web: una vez recibido el CSR, lo firma para obtener el certificado digital para el cliente (cliente.crt). El *Common Name* (CN) del certificado obtenido debe ser el nombre sin apellidos del alumno.
Nota: El *pass phrase* de la autoridad es "mica".
- Instalación del certificado en el cliente: obtención del archivo en formato PKCS12 que contenga tanto el certificado como la clave pública (cliente.p12).

Seguidamente, se utilizará dicho archivo para instalar el certificado con su clave privada en el navegador del cliente.

Al entregar la práctica se incluirán los cuatro archivos mencionados (cliente.key, cliente.crs, cliente.crt y cliente.p12) en la carpeta *certificados*.

12. En este apartado se utilizará el certificado digital para realizar autenticación del cliente. Para ello, se ampliará *login.php* de manera que permita al usuario autenticarse de dos formas distintas: a través del formulario web y a través de su certificado digital:



Usuario:	<input type="text"/>
Password:	<input type="password"/>



[autenticación con certificado](#)

Cuando el usuario pulse el enlace “autenticación con certificado” accederá a *logincert.php*, que será el único recurso en el que se pide certificado personal al cliente. Para ello, se realizarán las siguientes tareas.

- Configurar Apache y/o ficheros *.htaccess* para que pida el certificado personal cuando se solicite *logincert.php* (sólo *logincert.php*). Se autenticará dicho certificado según el protocolo SSL.
- Comprobar en *logincert.php* que el cliente es un usuario registrado en la base de datos. Con este fin, se obtendrá el nombre desde el certificado (variable de entorno `SSL_CLIENT_S_DN_CN`). La autenticación será correcta si este nombre se encuentra en el campo *user* de algún registro en la tabla *usuarios*.
- En caso de autenticación positiva, se actualizará convenientemente `$_SESSION`, y se realizará una redirección hacia la página *MasterWeb.php*. Por el contrario, si la autenticación no es positiva, la redirección será a *noAuth.php* (las mismas tareas que en *login.php*).

Documento: explicar las modificaciones realizadas en *httpd-ssl.conf* y/o en ficheros *.htaccess* para solicitar certificado al cliente cuando accede a *logincert.php*.

Documento: explicar por qué la página *logincert.php* consigue acceder a las variables de entorno que contienen la información del certificado del cliente. Concretamente, a `SSL_CLIENT_S_DN_CN`.

Coloque nuevamente la herramienta *Burp Suite* como *proxy* entre el navegador y el servidor web. Por HTTPS, intente autenticarse utilizando su certificado personal, verá que no es posible.

Documento: Justifique el motivo por el que no es posible dicha autenticación.

Configure *Burp Suite* para que sea posible la autenticación con certificado de cliente. Utilice para ello la pestaña *User Options* de *Burp Suite*. Una vez configurado *Burp Suite* es posible que tenga que volver a arrancarlo para que funcione correctamente.

Documento: Explique la configuración realizada en *Burp Suite*.