Tema 5. Desarrollo de aplicaciones Web utilizando código embebido

DESARROLLO EN ENTORNO SERVIDOR 2º DAW

MARÍA CRIADO DOMÍNGUEZ

Índice

- Autentificación de usuarios. Mecanismos.
- Sesiones.
- Cookies. Aplicaciones prácticas. Las cookies y el navegador.
- Seguridad: usuarios, perfiles, roles.

Autentificación de usuarios

Para identificar a los usuarios que visitan un sitio web, se pueden utilizar distintos métodos como el DNI digital o certificados digitales de usuario (documento digital que contiene información acerca del usuario como el nombre o la dirección. Esa información está firmada por otra entidad, llamada entidad certificadora, que debe ser de confianza y garantiza que la información que contiene es cierta), pero el más extendido es solicitar al usuario cierta información que solo él conoce: la combinación de un nombre de usuario y una contraseña.

Autentificación de usuarios

La información de autentificación (nombre y contraseña de los usuarios) se envía en texto plano desde el navegador hasta el servidor web. Esta práctica es altamente insegura y nunca debe usarse sin un protocolo como HTTPS que permita cifrar las comunicaciones con el servidor web.

Autentificación de usuarios
Mecanismos de autentificación

El protocolo HTTP ofrece un método sencillo para autentificar a los usuarios. El proceso es el siguiente:

- El servidor web debe proveer algún método para definir los usuarios que se utilizarán y cómo se pueden autentificar. Además, se tendrán que definir los recursos a los que se restringe el acceso y qué lista de control de acceso (ACL - lista de permisos sobre un objeto (fichero, directorio, etc.), que indica qué usuarios pueden utilizar el objeto y las acciones concretas que pueden realizar con el mismo (lectura, escritura, borrado, etc.)) se aplica a cada uno.
- Cuando un usuario no autentificado intenta acceder a un recurso restringido, el servidor web responde con un error de "Acceso no autorizado" (código 401).

Autentificación de usuarios
Mecanismos de autentificación

- El navegador recibe el error y abre una ventana para solicitar al usuario que se autentifique mediante su nombre y contraseña.
- La información de autentificación del usuario se envía al servidor, que la verifica y decide si permite o no el acceso al recurso solicitado. Esta información se mantiene en el navegador para utilizarse en posteriores peticiones a ese servidor.

Autentificación de usuarios

Mecanismos de autentificación

En el servidor web Apache, existe una utilidad en línea de comandos, htpasswd, que permite almacenar en un fichero una lista de usuarios y sus respectivas contraseñas. La información relativa a las contraseñas se almacena cifrada; aun así, es conveniente crear este fichero en un lugar no accesible por los usuarios del servidor web.

Ir a ruta /etc/apache2/

sudo htpasswd –c users dwes

-c solo con el primer usuario

el fichero se creara en la ruta /etc/apache2/users, que en principio no es accesible vía web.

Autentificación de usuarios
Mecanismos de autentificación

Para indicarle al servidor Apache qué recursos tienen acceso restringido, una opción es crear un fichero .htaccess en el directorio en que se encuentren, con las siguientes directivas:

```
AuthName "Contenido restringido"
AuthType Basic AuthUserFile /etc/apache2/users
require valid-user
```

Cambiar la configuración de apache2.conf

```
AllowOverride All
```

Autentificación de usuarios
Mecanismos de autentificación II

```
header('WWW-Authenticate: Basic Realm="Contenido restringido"');
```

Desde PHP puedes acceder a la información de autentificación HTTP que ha introducido el usuario utilizando el array superglobal $_SERVER. Valor Contenido

- $_SERVER['PHP_AUTH_USER'] Nombre de usuario que se ha introducido.
- $_SERVER['PHP_AUTH_PW'] Contraseña introducida.
- $_SERVER['AUTH_TYPE'] Método HTTP usado para autentificar. Puede ser Basic o Digest.

Si no introduces un usuario/contraseña válidos, el navegador te mostrará el error 401.

Además, en PHP puedes usar la función header para forzar a que el servidor envíe un error de "Acceso no autorizado" (código 401). De esta forma no es necesario utilizar ficheros .htaccess para indicarle a Apache qué recursos están restringidos.

Sesiones

- Una sesión es un mecanismo de programación de las tecnologías de web scripting que permite conservar información sobre un usuario al pasar de una página a otra.
- A diferencia de una cookie, los datos asociados a una sesión se almacenan en el servidor y nunca en el cliente.
- En el servidor web están almacenados todos los datos de la sesión y se accede a ellos cada vez que se pasa de página gracias al identificador almacenado en la cookie.

Sesiones

Directiva	Significado
session.use_cookies	Indica si se deben usar cookies (1) o propagación en la URL (0) para almacenar el SID.
session.use_only_cookies	Se debe activar (1) cuando utilizas cookies para almacenar los SID, y además no quieres que se reconozcan los SID que se puedan pasar como parte de la URL (este método se puede usar para usurpar el identificador de otro usuario).
session.save_handler	Se utiliza para indicar a PHP cómo debe almacenar los datos de la sesión del usuario. Existen cuatro opciones: en ficheros (`files`), en memoria (`mm`), en una base de datos SQLite (`sqlite`) o utilizando para ello funciones que debe definir el programador (`user`). El valor por defecto (`files`) funcionará sin problemas en la mayoría de los casos.
session.name	Determina el nombre de la cookie que se utilizará para guardar el SID. Su valor por defecto es `PHPSESSID`.
session.auto_start	Su valor por defecto es 0, y en este caso deberás usar la función `session start` para gestionar el inicio de las sesiones. Si usas sesiones en el sitio web, puede ser buena idea cambiar su valor a 1 para que PHP active de forma automática el manejo de sesiones.
session.cookie_lifetime	Si utilizas la URL para propagar el SID, éste se perderá cuando cierres tu navegador. Sin embargo, si utilizas cookies, el SID se mantendrá mientras no se destruya la cookie. En su valor por defecto (0), las cookies se destruyen cuando se cierra el navegador. Si quieres que se mantenga el SID durante más tiempo, debes indicar en esta directiva ese tiempo en segundos.
session.gc_maxlifetime	Indica el tiempo en segundos que se debe mantener activa la sesión, aunque no haya ninguna actividad por parte del usuario. Su valor por defecto es 1440. Es decir, pasados 24 minutos desde la última actividad por parte del usuario, se cierra su sesión automáticamente.

Sesiones

Inicio de sesión

1. Ejecutando la función **session_start()** para indicar a PHP que inicie una nueva sesión o reanude la anteriorTodas las páginas que necesiten utilizar la información almacenada en la sesión, deberán ejecutar la función session_start().
2. Si has activado la directiva session.auto_start en la configuración de PHP, la sesión comenzará automáticamente en cuanto un usuario se conecte a tu sitio web.
   - En caso de que ese usuario ya haya abierto una sesión con anterioridad, y ésta no se haya eliminado, en lugar de abrir una nueva sesión se reanudará la anterior.
   - Si has activado la directiva session.auto_start en la configuración de PHP, la sesión comenzará automáticamente en cuanto un usuario se conecte a tu sitio web.
   - En caso de que ese usuario ya haya abierto una sesión con anterioridad, y ésta no se haya eliminado, en lugar de abrir una nueva sesión se reanudará la anterior.

Sesiones

session_id(): asigna un identificador a una sesión SID (se recomienda que sea corto y que contenga sólo caracteres alfanuméricos).Si no asignamos un identificador, se asignará uno automáticamente. En caso de ser usada, debe estar antes de session_start().

session_name(): asigna/retorna el nombre a una sesión (se recomienda que sea corto y que contenga sólo caracteres alfanuméricos). Si no indicamos un nombre, se usará por defecto el definido en el archivo de configuración de PHP (por defecto es **PHPSESSID**).

IMPORTANTE: en caso de usarse deberá estar definida en cada archivo .php, y situada antes de session_start().

SID Identificador de sesión

- Cada usuario distinto de un sitio web tiene su propia información de sesión. Para distinguir una sesión de otra se usan los identificadores de sesión (SID).
- Un SID es un atributo que se asigna a cada uno de los visitantes de un sitio web y lo identifica. Es único para cada usuario.
- De esta forma, si el servidor web conoce el SID de un usuario, puede relacionarlo con toda la información que posee sobre él, que se mantiene en la sesión del usuario.

Variable superglobal $_SESSION

Mientras la sesión permanece abierta, puedes utilizar la variable superglobal $_SESSION para añadir información a la sesión del usuario, o para acceder a la información almacenada en la sesión.

Cerrar una sesión

Cerrar una sesión supone:

- Destruir la matriz $_SESSION
- Borrar el identificador de la sesión (SID).

Puede cerrarse:

- El usuario: puede cerrar la sesión simplemente cerrando el navegador.
- Un script: puede cerrar la sesión mediante la función session_destroy().
- El servidor: puede cerrar la sesión cuando ha pasado el tiempo indicado en segundos en la directiva session.gc_maxlifetime.

Cookies

Una cookie es un fichero de texto que un sitio web guarda en el entorno del usuario del navegador. Su uso más típico es el almacenamiento de las preferencias del usuario (por ejemplo, el idioma en que se deben mostrar las páginas), para que no tenga que volver a indicarlas la próxima vez que visite el sitio.

Según el protocolo HTTP, las cookies no pueden ser más grandes de 4096 Bytes (4KB).

También hay un límite en el número total de cookies en el disco duro del cliente. Suele ser de unas 300 cookies. Cuando se llega a este número, una cookie antigua se elimina antes de crear la nueva.

Cookies

Los servidores web sólo pueden acceder a cookies establecidas por su propio dominio.

Este dominio es establecido por el navegador cuando el servidor crea una nueva cookie , y sólo puede ser un dominio o subdominio del servidor.

Según su duración:

- **Cookies de sesión**

Las cookies de sesión son cookies cortas que caducan al cerrar una página web, por ejemplo, al cerrar la pestaña o el navegador/app.

- **Cookies persistentes**

Opuestas a las cookies de sesión, las cookies persistentes son aquellas que siguen almacenadas en el ordenador durante más tiempo incluso después de la sesión (pueden ser horas, meses o años).

Cookies

Según su finalidad:

- **Cookies de técnicas**
- **Cookies personalización**
- **Las cookies de análisis**
- **Las cookies publicitarias**
- **Las cookies de redes sociales y otros plugings**

Cookies

En PHP, para almacenar una cookie en el navegador del usuario, puedes utilizar la función setcookie. El único parámetro obligatorio que tienes que usar es el nombre de la cookie, pero admite varios parámetros más opcionales. http://es.php.net/manual/es/function.setcookie.php

Por ejemplo, si quieres almacenar en una cookie el nombre de usuario que se transmitió en las credenciales HTTP puedes hacer:

setcookie("nombre", “maría”, time()+3600);

Los dos primeros parámetros son el nombre de la cookie y su valor. El tercero es la fecha de caducidad de la misma (una hora desde el momento en que se ejecute). En caso de no figurar este parámetro, la cookie se eliminará cuando se cierre el navegador. Ten en cuenta que también se pueden aplicar restricciones a las páginas del sitio que pueden acceder a una cookie en función de la ruta.

Cookies

Por ejemplo, si quieres almacenar en una cookie el nombre de usuario que se transmitió en las credenciales HTTP (es solo un ejemplo, no es en absoluto aconsejable almacenar información relativa a la seguridad en las cookies), puedes hacer:

setcookie("nombre", “maría”, time()+3600);

Los dos primeros parámetros son el nombre de la cookie y su valor. El tercero es la fecha de caducidad de la misma (una hora desde el momento en que se ejecute). En caso de no figurar este parámetro, la cookie se eliminará cuando se cierre el navegador. Ten en cuenta que también se pueden aplicar restricciones a las páginas del sitio que pueden acceder a una cookie en función de la ruta.

Cookies

Las cookies se transmiten entre el navegador y el servidor web de la misma forma que las credenciales que acabas de ver; utilizando los encabezados del protocolo HTTP. Por ello, las sentencias setcookie deben enviarse antes de que el navegador muestre información alguna en pantalla.

El proceso de recuperación de la información que almacena una cookie es muy simple. Cuando accedes a un sitio web, el navegador le envía de forma automática todo el contenido de las cookies que almacene relativas a ese sitio en concreto.

Desde PHP puedes acceder a esta información por medio del array $_COOKIE.

Cookies

Es importante tener en cuenta al trabajar con cookies es el orden en que se realiza el envío y la creación de cookies, así como su disponibilidad en $_COOKIES:

- Cuando una página pide al navegador que cree una cookie, el valor de la cookie no está disponible en $_COOKIE en esa página en ese momento.
- El valor de la cookie estará disponible en $_COOKIE en las páginas en llamadas posteriores, cuando el navegador las pida al servidor y envíe el valor de la cookie en la petición.

Cookies

Para borrar una cookie, simplemente se debe volver a crear la cookie con un tiempo de expiración anterior al presente.

```
setcookie("nombre", "Pepito Conejo", time() - 1);
```

Si solamente queremos borrar el valor almacenado en la cookie sin borrar la propia cookie, simplemente se debe volver a crear la cookie, sin indicarle el valor a almacenar:

```
// Esta cookie no se borra, pero no guardará ningún valor.
  setcookie("nombre");
```