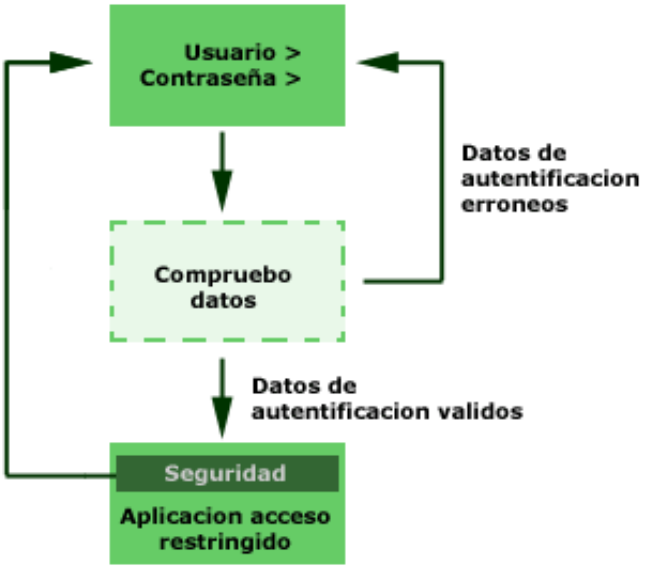
Un sistema de Autentificación en PHP

Volver

Funcionamiento del sistema de autentificación en PHP

Un sistema de autentificación es un módulo de seguridad para asegurarnos de que el usuario que visita las páginas es quien dice ser. Por supuesto, sabiendo que ese usuario es conocido, podremos darle acceso a más aspectos de la página que si fuese un usuario desconocido. Pero supongo que, si estás leyendo este artículo, ya conocerás lo que es un sistema de autentificación y lo que deseas hacer es crear uno para tus páginas.

Esquema de un sistema de autentificación
(Ejecucion) (Pincha en los modulos del grafico para ver el codigo PHP de cada uno)



Vamos a empezar por definir un diagrama para realizar la autentificación de usuario en unas páginas web, que nos servirá para programar luego las páginas ajustándose al diagrama.

En la imagen anterior podemos ver el diagrama, que empieza por la página donde se pide un usuario y contraseña para acceder a la aplicación de acceso restringido.

Los datos de autentificación (usuario y contraseña escritos en la página inicial) se envían a la página dibujada con línea de puntos, que se encarga de hacer una comprobación de dichos datos del usuario. Según los datos de autentificación, se redirecciona al navegador a la página de la aplicación restringida, en caso de que sean correctos, o a la página donde volver a escribir el usuario/contraseña, en caso de que sean incorrectos. Esta página la he dibujado con línea de puntos porque no es una página donde se pare el navegador para nada, sino que sólo es una página de paso que redirecciona a un sitio u otro dependiendo de los datos que reciba.

La aplicación de acceso restringido, aparte de mostrar las funcionalidades que queríamos proteger con usuario contraseña, debe de realizar unas comprobaciones de seguridad para saber si se ha pasado con éxito el proceso de autentificación o si se está intentando acceder de manera no permitida a esa página. Esta comprobación la he dibujado como una capa con color verde más oscuro sobre la página de la aplicación. Si no se satisface dicha comprobación (el usuario no se ha autentificado correctamente) se vuelve a la página donde escribir el usuario y la contraseña.

Este es el esquema básico, que espero que se entienda bien. Ahora, veamos algunas preguntas que podría hacerse el lector.

¿Por qué hacemos esta comprobación de seguridad dentro de la aplicación?
Podría ser que alguien conociese la URL de la aplicación de acceso restringido y la escribiese directamente sobre la barra de direcciones del explorador, así que hacemos esta comprobación para saber que realmente no se está accediendo sin pasar por la página que comprueba si el usuario/contraseña es correcto

¿Cómo sabemos que ciertamente se ha pasado por la página que comprueba los datos de autentificación?
Esta comprobación la podríamos hacer de varias maneras, así pues, depende de nuestro script de autentificación y el nivel de seguridad que tratemos de implementar. Un ejemplo simple podría ser crear una variable de sesión en la página que comprueba los datos, si es que eran correctos, y en capa se seguridad de las páginas de acceso restringido comprobaríamos si esa sesión está o no definida.

¿Cómo pueden entrar a esa página si no hay enlaces directos y para pasar a ella necesitamos que nos redireccione la página de comprobación del usuario/contraseña?

Pues de diversas maneras, para empezar, el historial de los ordenadores guarda las URL a las que se ha accedido y cualquier persona podría recuperar la URL de nuestra aplicación con acceso restringido. También se podría probar distintas URL que podríamos imaginarnos como posibles para la aplicación y esperar a acertar con el nombre de archivo en algún momento, incluso esta tarea se la podríamos encomendar a un programa para realizar muchas más pruebas. En cualquier caso, nuestra seguridad no se puede quedar en simplemente que los posibles intrusos no conozcan la dirección de la página.

Página inicial con el formulario de autentificación en PHP

Vamos a realizar la página inicial, que tiene el formulario de autentificación en el que el visitante debería rellenar con su usuario y contraseña. Como es la página inicial, la llamaremos index.php, que es el documento por defecto configurado en nuestro servidor.

Para realizar esta página, utilizaremos HTML básico, excepto en una comprobación que nos permitirá saber si se accede al formulario de nuevo por no haber introducido correctamente el usuario y contraseña, pues, en ese caso, habría que mostrar un cartelito informando que el usuario o la contraseña no son válidos.

Para pasar a la página inicial el mensaje de que el usuario/contraseña introducidos no son válidos utilizaremos una variable pasada a través de la URL. La llamaremos errorusuario, y si contiene la cadena "si" es que estamos recibiendo un error.

El código sería el siguiente:

```
<html>
<head>
<title>Autentificación PHP</title>
</head>
<body>
<h1>Autentificación PHP</h1>
<form action="control.php" method="POST">
<table align="center" width="225" cellspacing="2" cellpadding="2" border="0">
<tr>
<td colspan="2" align="center"
<?if ($_GET["errorusuario"]=="si"){?>
bgcolor=red><span style="color:ffffff"><b>Datos incorrectos</b></span>
<?}else{?>
bgcolor=#cccccc>Introduce tu clave de acceso
<?}?></td>
</tr>
<tr>
<td align="right">USER:</td>
<td><input type="Text" name="usuario" size="8" maxlength="50"></td>
</tr>
<tr>
<td align="right">PASSWD:</td>
<td><input type="password" name="contrasena" size="8" maxlength="50"></td>
</tr>
<tr>
<td colspan="2" align="center"><input type="Submit" value="ENTRAR"></td>
</tr>
</table>
</form>
</body>
</html>
```

> **Nota:** La variable errorusuario, recibida por la URL y que informa si se produjo un error anterior al introducir usuario y contraseña, se está recogiendo por mediación del array asociativo $_GET, que guarda todas las variables enviadas por la URL.

El formulario tiene el atributo action dirigido hacia la página "control.php", que es la que se encarga de recoger los datos y ver si son correctos. Será tratada en el próximo capítulo.

Control de los datos de autentificación en PHP

Esta página será encargada de decidir si los datos de configuración son correctos y actuar en consecuencia. Dependiendo del nivel de seguridad que queramos aplicar a nuestra aplicación, esta página será más o menos complicada.

En un principio no deseo liar mucho las cosas, así que explicaré una versión muy reducida de este archivo de control, en la que se comprueba si el usuario y contraseña sean dos valores específicos. Esto tiene la desventaja que sólo podemos crear un usuario/contraseña distinto y no un sistema que permita muchos usuarios distintos. Bueno, en realidad si que permitirá que accedan muchos usuarios a la vez, pero utilizando todos el mismo nombre de usuario y contraseña.

En aplicaciones más avanzadas podríamos tener en una base de datos una lista de usuarios con sus contraseñas. Entonces, en este archivo de control deberíamos hacer una búsqueda para ver si existe una correspondencia en la base de datos de ese usuario con esa contraseña. Esto lo veremos en adelante, ahora nos quedamos con la versión reducida.

Después de la comprobación podrán pasar dos cosas:
Si los datos son correctos, definirá una variables de sesión que servirá para saber que ese visitante ha sido validado correctamente y tiene permiso para acceder a la aplicación. Además redireccionará al visitante a la página de la aplicación restringida.
Si el usuario/contraseña no era correcto, se envía al navegador a la página de inicio pasando la variable errorusuario=si, que indica que ha habido un error en la autentificación.

El código se puede ver a continuación:

```
<?
//vemos si el usuario y contraseña es váildo
if ($_POST["usuario"]=="usuario" && $_POST["contrasena"]=="password"){
    //usuario y contraseña válidos
    //defino una sesion y guardo datos
    session_start();
    session_register("autentificado");
    $autentificado = "SI";
    header ("Location: aplicacion.php");
}else {
    //si no existe le mando otra vez a la portada
    header("Location: index.php?errorusuario=si");
}
?>
```

Referencia: Dos enlaces a documentación relacionada en nuestro manual de Programación en PHP:

Capa de seguridad en PHP

Este archivo, en nuestro caso llamado seguridad.php, se encargará de dotar seguridad a toda la aplicación de acceso restringido. La técnica que vamos a utilizar es incluirlo al principio de todas las páginas que queramos que permitan un acceso restringido.

El módulo de seguridad, incluido al principio de cada archivo, realizará las comprobaciones oportunas y actuará permitiendo ver el archivo o denegando su visualización dependiendo de dichas comprobaciones.

Dependiendo del nivel de seguridad que deseemos implementar, la creación de este archivo puede ser más o menos complicada. Como no deseo complicar en un principio los scripts, esta versión resultará bastante sencilla.

Lo único que haré será recuperar la variable de sesión donde guardo si ese usuario ha sido autentificado o no. Luego se comprueba esa variable para saber si se ha autentificado el usuario o no, realizando estas acciones:

Si no se había autentificado, redirijo al navegador a la página que tiene el formulario de autentificación. Además, salgo del script PHP, con lo que la página deja de ejecutarse y el resto no se verá. Sólo se mandará al navegador la redirección con lo que el navegador se moverá la formulario y será imposible ver nada en la página segura.

Si se había autentificado, no hago nada. Ni tan siquiera trato este caso, de modo que se seguiría ejecutando la página con el contenido que correspondiese. No hay que olvidar que este archivo de seguridad se va a ejecutar como un include al principio de todos los archivos de la aplicación restringida, lo que significa que, si no se hace nada, se seguiría mostrando la página donde este archivo está incluido.

El código se puede ver a continuación:

```
<?
//TOMO VARIABLES DE SESION SOBRE LA AUTENTIFICACION
session_register("autentificado");
//COMPRUEBA QUE EL USUARIO ESTA AUTENTIFICADO
if ($autentificado != "SI") {
    //Si autntificado es distinto de "SI", envio a la página de autentificacion y no ejecuto mas del archivo .php
    header("Location: index.php");
    //ademas salgo de este script
    exit();
}
?>
```

Archivos de la aplicación con acceso restringido en PHP

La aplicación con acceso restringido se realizará como cualquier otra aplicación de PHP, con la salvedad de que, a todos los archivos que queramos proteger, habrá que incluirles al principio la capa de seguridad, representada por el archivo seguridad.php.

Como decía, todo en el archivo de la aplicación se realizará como cualquier otro archivo de PHP, es decir, con sólo incluir el módulo de seguridad, el archivo ya tendrá el acceso restringido y todo lo demás lo haremos de manera transparente a este estado de seguridad.

El código de una página segura sería el siguiente:

```
<?include ("seguridad.php");?>
<html>
<head>
<title>Aplicación segura</title>
</head>
<body>
<h1>Si estás aquí es que te has autentificado</h1>
<br>
----
<br>
Aplicación segura
<br>
----
<br>
<br>
<a href="salir.php">Salir</a>
</body>
</html>
```

> **Importante:** El include del archivo seguridad.php se ha de realizar en la primera línea del archivo PHP de la aplicación. Si no lo hacemos en la primera línea o si escribimos texto en la página antes de incluir la capa de seguridad, el script podría fallar y hacer que no funcione la aplicación o que sea menos segura. Este efecto se produce porque no se puede escribir en la página nada si se desea hacer una redirección con PHP (función header) y si se escribe algo, la redirección no podrá funcionar.

Un detalle incluido es un enlace para salir de la aplicación, que se dirige a el archivo salir.php.

Salir de la aplicación segura en PHP

La seguridad de la aplicación se basa en la definición de unas variables de sesión que se consultan en cada página segura. Puede ocurrir que el usuario entre en la aplicación e inicie una sesión y que se marche de la aplicación segura sin cerrar la sesión, con lo que quedaría abierta para que cualquier otra persona pueda acceder a la aplicación volviendo por el historial de páginas del navegador.

Las sesiones se finalizan solas cuando pasa un determinado tiempo sin recibir nuevas peticiones, pero no deseamos que antes de que se finalicen se pueda acceder con ese ordenador a nuestra aplicación restringida.

Parece interesante, pues, ofrecer al visitante la opción de acabar la sesión en cualquier momento, para asegurarnos en ese caso que la sesión se ha terminado y no se podrá acceder si no es introduciendo nuevamente el usuario y contraseña correctos.

El archivo en concreto lo único que hace es terminar la sesión asociada a su acceso. Podemos ver el código a continuación.

```
<?
session_start();
session_destroy();
?>
<html>
<head>
<title>Has salido!!</title>
</head>
<body>
Gracias por tu acceso
<br>
<br>
<a href="index.php">Formulario de autentificación</a>
</body>
</html>
```

Autentificación PHP para múltiples usuarios usando MySQL

Vamos a ver las páginas PHP que necesitaríamos para realizar un acceso restringido por clave y contraseña para múltiples usuarios, donde cada uno tenga unos datos de acceso propios.

La base de datos

La base de datos que vamos a utilizar contendrá una tabla para los usuarios, donde cada uno dispondrá, al menos, de dos campos: un nombre de usuario y una contraseña, los dos de tipo texto.

Tabla usuario

Nombre del campo	Tipo del campo
nombre_usuario	Texto
clave_usuario	Texto

En una base de datos de usuarios, el nombre de usuario debería ser un valor único, irrepetible para otro usuario, es decir, no podremos tener dos usuarios con el mismo nombre. Por esta razón, el campo nombre_usuario podría ser la clave principal de la tabla, aunque también podríamos haber creado un campo adicional, llamado por ejemplo id_usuario, de tipo autonimérico y colocarlo como clave principal.

Para conseguir no insertar dos usuarios con el mismo nombre de usuario, a la hora de insertarlos en la tabla, comprobaremos que no haya ningún usuario ya introducido con el nombre de usuario que se pretende insertar. Este paso, aunque importante, no lo vamos a ver, pues sólo nos vamos a centrar en decidir si un usuario puede entrar o no en la aplicación, suponiendo que los usuarios se encuentran ya insertados en la base de datos.

En el ejemplo suponemos que utilizamos una base de datos MySQL, sin embargo, cualquier tipo de base de datos podrá servir para unos objetivos como los que nos proponemos.

El funcionamiento del script

El script que se utilizará para decidir si un usuario puede o no entrar en la aplicación es muy sencillo. Simplemente hace una llamada a la base de datos para comprobar si los datos de autentificación escritos por el visitante (usuario y contraseña) corresponden con los de algún usuario. En caso de que así sea, se permite la entrada y de no ser así, se deniega.

Lo primero sería abrir una conexión con la base de datos y seleccionar la base con la que hemos de trabajar.

```
//se conecta con la base de datos
$conn = mysql_connect("servidor","usuario","password");
//selecciono la BBDD
mysql_select_db("nombre_bbdd",$conn);
```

Un segundo paso es construir una sentencia SQL que nos permita comprobar si existe o no un usuario con los datos de autentificación introducidos. Utilizamos una simple sentencia SELECT, sobre la tabla de usuarios, donde se extraen usuarios que tengan el mismo nombre de usuario y la contraseña introducidos en la página de acceso.

```
//Sentencia SQL para buscar un usuario con esos datos
$ssql = "SELECT * FROM usuario WHERE nombre_usuario='$usuario' and clave_usuario='$contrasena'";

//Ejecuto la sentencia
```

```
$rs = mysql_query($ssql,$conn);
```

Si esa sentencia SELECT responde con algún registro encontrado, sabremos que existe un usuario donde sus datos de autentificación corresponden perfectamente con los introducidos. En ese caso podremos realizar las acciones encaminadas a permitir el acceso. Por el contrario, si la sentencia SELECT no encuentra ningún registro, sabremos que no existe un usuario con los datos de autentificación introducidos y por lo tanto, deberemos realizar las acciones encaminadas a restringir el acceso.

```
if (mysql_num_rows($rs)!=0){
   //usuario y contraseña válidos
   //defino una sesion y guardo datos
   session_start();
   session_register("autentificado");
   $autentificado = "SI";
   header ("Location: aplicacion.php");
}else {
   //si no existe le mando otra vez a la portada
   header("Location: index.php?errorusuario=si");
}
```

Las acciones para restringir o permitir el acceso son exatamente iguales a las que veníamos utilizando en el script de control sin utilizar la base de datos. Así que no vamos a comentarlas más, sino que os referimos al artículo donde las explicamos.

El código completo del ejemplo sería el siguiente.

```
<?
//conecto con la base de datos
$conn = mysql_connect("servidor","usuario","password");
//selecciono la BBDD
mysql_select_db("nombre_bbdd",$conn);

//Sentencia SQL para buscar un usuario con esos datos
$ssql = "SELECT * FROM usuario WHERE nombre_usuario='$usuario' and clave_usuario='$contrasena'";

//Ejecuto la sentencia
$rs = mysql_query($ssql,$conn);

//vemos si el usuario y contraseña es váildo
//si la ejecución de la sentencia SQL nos da algún resultado
//es que si que existe esa conbinación usuario/contraseña
if (mysql_num_rows($rs)!=0){
   //usuario y contraseña válidos
   //defino una sesion y guardo datos
   session_start();
   session_register("autentificado");
   $autentificado = "SI";
   header ("Location: aplicacion.php");
}else {
   //si no existe le mando otra vez a la portada
   header("Location: index.php?errorusuario=si");
}
mysql_free_result($rs);
mysql_close($conn);
?>
```

> **Nota:** Es importante destacar que esta página no debería contener ningún tipo de texto antes de la apertura de código PHP, ni tan siquiera saltos de línea. Esto es debido a que al final se realiza una redirección y este tipo de instrucciones solamente se puede ejecutar si no se ha escrito todavía ningún carácter en el cuerpo.
