GUIA N° 11 – SEGURIDAD Y SESIONES EN PHP

Objetivo:

- ✓ Conocer y utilizar correctamente sesiones para proporcionarle seguridad a una aplicación WEB desarrollada con PHP

I. ¿QUÉ ES UNA SESIÓN?

Una sesión en una web se puede ver como el recorrido de páginas que un usuario hace en un sitio, desde que entra hasta que lo abandona. Gracias al uso de sesiones podemos reconocer las peticiones de cada usuario y así llevar a cabo acciones específicas, como mostrar información adaptada a él o guardar información de sus gustos o páginas que más visita.

El uso de sesiones proporciona una forma fácil de guardar datos del usuario de forma temporal para reutilizarlos en el momento que se necesiten. Las sesiones son muy utilizadas para saber si un usuario a ingresado a través de login (usuario + contraseña) y en cualquier cosa que requiera una variable temporal en relación a la visita del usuario.

De forma general pueden diferenciarse dos tipos de sesiones:

- ✓ Sesiones “activas”: son las que muestran información personalizada según el usuario, por ejemplo, cuando inicia sesión en una web.

- ✓ Sesiones “pasivas”: son aquellas donde el servidor reconoce cada movimiento del usuario y lo almacena de forma que en un futuro, se le mostrará una web con la información que al usuario le pueda parecer más interesante sin que se dé cuenta.

II. ¿CÓMO FUNCIONA UNA SESIÓN?

Las sesiones se basan en un identificador generado por visitante, simulando una especie de DNI (Identificador Único), de modo que cuando quiera acceder a cualquier página de un sitio, mostrará ese “DNI” y quedará identificado. El identificador será único por cada usuario y se le conoce como Sesión ID (SID).

Los Sesión ID (SID) se gestionan por medio de las denominadas Cookies. Una cookie es información que se establece desde el servidor y que el navegador del usuario envía en cada petición. Por lo tanto, si se establece en una cookie el identificador de sesión, se tiene entonces una tarjeta identificadora lista para usar.

Desde el lado del servidor, se debe implementar toda la lógica de la sesión, almacenando todos los SID activos y la información relativa a ellos. Esto se puede hacer de múltiples formas: mediante sistema de ficheros, base de datos o cualquier otro método de almacenamiento de información.

III. SESIONES EN PHP

PHP tiene todo una serie de funciones para el manejo de sesiones que facilitan en gran medida la labor del desarrollador. Estas funciones llevarán a cabo tareas tales como:

- ✓ Identificación por cookie del SID (Si no existe el SID, crea uno y lo guarda en las cookies).
- ✓ Almacenamiento de información relativa al SID en el sistema de ficheros.
- ✓ Gestión del expirado o finalización de sesión.

En el manejo de sesiones en PHP se deben tener en cuenta los siguientes elementos:

1. ***SESSION_START():*** Las sesiones en PHP se crean llamando la función "session_start()" de la siguiente forma:

```php
<?php
  session_start();
 ?>
```

Llamando esta función se inicia una sesión. Para poder usar las variables de sesión creadas hay que colocar esta línea al principio de nuestros script en PHP antes de cualquier sentencia HTML o se generará un error.

2. ***$_SESSION:*** En PHP para almacenar y/o recuperar información sobre la sesión actual se usa un array global llamado $_SESSION. Para definir una variable de sesión en PHP (que se guarda de manera temporal) se utiliza:

```php
<?php
  $_SESSION["Varible_De_Sesion"]="valor";
?>
```

3. ***HEADER:*** La forma más sencilla de realizar un redireccionamiento (redirect) en PHP es indicarle al navegador en las cabeceras de respuesta que debe llevar al usuario a otra página.

En PHP podemos enviar esta información con el uso de la función header, por ejemplo:

- ✓ `Header ('Location: http://www.google.com/');`
- ✓ `Header ('Location: index.php');`

El primer ejemplo llevaría al usuario al buscador de google y el segundo lo llevaría a una página llamada index.php. Lo importante a tener en cuenta es que la función header se debe utilizar antes de que se imprima cualquier otra información de la página. Es decir no debe haber ni echos ni prints ni ninguna salida como bloques de html, sino la redirección no funcionará correctamente

4. ***SESSION_DESTROY():*** Para finalizar una sesión PHP se utiliza la función "session_destroy():

```php
<?php
   session_destroy();
?>
```

Una excelente forma de implementar la finalización de una sesión en PHP es creando un página totalmente independiente (por ejemplo: Salir.php) que al ser llamada por medio de un link ejecute un código similar al que se muestra a continuación:

```php
<?php
     session_start();
     session_destroy();
     header("Location:login.php");
?>
```

Donde "login.php" es la página que cargará impedientemente sea finalizada la sesión.

5. ***EXIT():*** Al llamar esta función se finaliza un script php inmediatamente, es decir, cualquier programa escrito en lenguaje PHP termina su ejecución.

Es recomendable que a cada página perteneciente a un sitio Web PHP quede protegida de cualquier acceso no autorizado sin haber iniciado una sesión con un usuario y una contraseña. Con el fin de proporcionarle este nivel de seguridad a las páginas que así lo requieran se debe agregar al inicio del código PHP de cada una de ellas líneas de código similares a las que se observan a continuación:

```php
<?php
  session_start();
  if($_SESSION["autenticado"]!="1")
  {
    Header("Location:login.php");
    exit();
  }
    .
    .
    .
    .

?>
```

Donde `$_SESSION["autenticado"]` es una variable de sesión cuyo valor se ha asignar en el formulario de "Inicio de Sesión" en el instante que el usuario escriba su nombre de usuario y su contraseña e intente acceder a la aplicación.

.

IV. EJEMPLO "INCIO DE SESION EN PHP"

El siguiente código muestra un ejemplo de cómo implementar un Inicio de Sesión en PHP. Tenga en cuenta que este ejercicio requiere de un tabla en una base de datos que contenga la información de nombres de usuario y contraseñas, un procedimiento almacenado que realice la consulta de estos datos y una clase ("usuario.php") con el método ("Consultarusuarios") que llame a dicho procedimiento.

Estos elementos no se muestran en este código, solo se muestra el código correspondiente a la página WEB ("login.php") donde se le solicita el nombre de usuario y su contraseña a un usuario para que inicie una sesión en la aplicación.

```
<HTML>
<HEAD>
   <TITLE>LOGIN</TITLE>
</HEAD>
<BODY>
<FORM ACTION="login.php" NAME="index" METHOD="POST">
<h2>iniciar sesion</h2>
Usuario:
<INPUT TYPE="text" NAME="nombre" SIZE="20">
Contraseña:
<INPUT TYPE="password" NAME="contrasena" SIZE="20">
<br>
<INPUT TYPE="SUBMIT" NAME="entrar" VALUE="entrar">
<br>
Si no te has registrado regístrate <a href="registro.php">aqui</a>
</FORM>

<?php
session_start();

if (isset($_POST['entrar']))
{
  include ("Php_Code/usuario.php");
  $Objusuario = new usuario();
  $nombre = $_POST['nombre'];
  $contrasena = $_POST['contrasena'];

  $consulta = $Objusuario->Consultarusuarios($nombre,$contrasena);
  $numfilas = mysql_num_rows($consulta);

if ($numfilas> 0)
{
  $_SESSION["autenticado"]="1";
  $_SESSION["nombreusuario"]=$_POST["nombre"];
  header("Location:index.php");
}
else
{
  $_SESSION["autenticado"]="0";
  echo "usuario y o clave incorrecta";
}
}
?>
```