Guía Práctica de Ejercicios Nº 12

En nuestra ante última práctica veremos conceptos de seguridad en aplicaciones Web, principalmente desarrolladas en PHP. Pero también veremos la manera de configurar nuestro lenguaje por medio del archivo php.ini que define las capacidades del mismo.

Objetivo: Comprender los conceptos básicos y medios de seguridad Web con PHP como lenguaje del lado del servidor. Definir, desarrollar y utilizar los métodos de seguridad básicos para proteger nuestra aplicación Web a posibles ataques de tipo XSS, SQL Injection, Remote Code, Command Injection, session fixation, session hijacking, entre otros. Desarrollar e implementar el filtrado de datos en nuestra aplicación Web. Entender y configurar las capacidades del lenguaje PHP desde un script o utilizando el archivo php.ini.

Introducción: En nuestra actualidad vemos que el desarrollo de Internet ha sido inminente y con ello las aplicaciones Web, por lo tanto, se hace indispensable el uso de un lenguaje que permita desarrollar aplicaciones Web como PHP, entre otros. Teniendo en cuenta la situación anterior es que veremos la como desarrollar aplicaciones Web que se ejecutarán del lado del servidor utilizando uno de los mejores lenguajes del ambiente del Software Libre. Teniendo en cuenta la magnitud del proyecto a desarrollar veremos la necesidad de emplear el paradigma orientado a objetos con PHP; que nos dará la posibilidad de diseñar, desarrollar y mantener el Software de manera profesional utilizando un paradigma antes mencionado.

Construcciones (palabras clave), variables globales y funciones que estudiaremos en esta unidad.

Construcciones del lenguaje con las que trabajaremos.

session_regenerate_id regenera el identificador de la sesión.

md5 nos permite calcular el hash md5 de una cadena.

htmlentities convierte todos los caracteres a su entidad HTML aplicable.

htmlspecialchars convierte caracteres especiales a entidades HTML.

init_set nos permite establecer una nueva configuración de alguna capacidad del lenguaje.

init_get nos permite obtener la configuración de alguna capacidad del lenguaje.

init_get_all nos permite obtener la configuración de todas las capacidades del lenguaje.

Ejercicios

Utilizar los conceptos de programación para otorgar seguridad a nuestras aplicaciones

1) – Utilice los scritps de la clase anterior donde se desarrollo un formulario de login. Tenga en cuenta que deberá tener creada una base de datos y una tabla con usuarios de acceso (utilice la herramienta phpmyadmin para crear la base de datos y la tabla). Escriba el código necesario para mitigar problemas de ataques de sesión fixation al formulario de login.

db.php

```
<?php

$host = '127.0.0.1';
$user = 'root';
$password = '';
$db = 'sgu';

$enlace = mysql_connect('127.0.0.1', 'root', 'asdasd');
mysql_select_db('sgu', $enlace);
```

login.php

```
<?php
// Se inicia o reanuda una sesión
session_start();

// Se agrega el formulario de login
echo "<form action='proceso.php' method='post'>";
// Se genera el token para evitar algunos ataques
$token = md5(uniqid(rand(), true));
// Se guarda el token en la sesión
$_SESSION['token'] = $token;
// Se guarda el token como un componente oculto en el formulario
echo "<input type='hidden' name='token' value=".$token." />";
// Se crea una tabla con los elementos del formulario
echo "<table>";
echo "<tr>";
echo "<td>";
echo "Nombre de usuario: ";
echo "</td>";
echo "<td>";
echo "<input type='text' name='username' id='username'>";
echo "</td>";
echo "</tr>";
echo "<tr>";
echo "<td>";
echo "Contrase&ntilde;a: ";
echo "</td>";
echo "<td>";
```

```
echo "<input type='password' name='password' id='password'>";
echo "</td>";
echo "</tr>";
echo "<tr>";
echo "<td>";
echo "<input type='submit' name='entrar' id='entrar' value='Entrar'>";
echo "</td>";
echo "<td>";
echo "</td>";
echo "</tr>";
echo "</table>";
echo "</form>";
```

proceso.php

```
<?php
// Se inicia o reanuda una sesión
session_start();

// Se requiere del archivo db.php
require_once 'db.php';

// Si fue enviado el formulario se procesará
if (isset($_POST['entrar']) && $_POST['entrar'] == 'Entrar' && isset($_POST['token']) &&
$_POST['token'] == $_SESSION['token']) {
	// Se verifican los datos de login
	if (isset($_POST['username'])) {
		$username = htmlentities(trim($_POST['username']));
	}
	if (isset($_POST['password'])) {
		$password = htmlentities(trim($_POST['password']));
	}

	$consulta = "SELECT * FROM Usuarios WHERE username =
'".mysql_escape_string($username)."' AND password = '".mysql_escape_string($password)."'";
	$resultado = mysql_query($consulta);
	// Se evalua si el usuario existe y se encuentra habilitado
	if (mysql_num_rows($resultado) > 0) {
		// Registrar variables de sesiÃ³n
		$_SESSION['usuarioRegistrado'] = true;
		$_SESSION['username'] = $username;
		// Se regenera el id de la sesión
		session_regenerate_id();
		// Se direcciona a la página de aplicaciones
		header("Location: loginok.php");
	} else {
		// Se hace una redirección por un mal login
		header("Location: loginfail.php");
	}
}
```

loginok.php

```
<?php
```

```
session_start();

echo "Login Ok!!!";
echo "<br/>";
echo "Usuario: ".htmlentities($_SESSION['username']);
```

loginfail.php

```
<?php
session_start();

echo "Login Fail!!!";
```

2) – Del ejercicio anterior ahora deberá mitigar posibles ataques de sesión hijacking al formulario de login.

proceso.php

```
<?php
// Se inicia o reanuda una sesión
session_start();

// Se requiere del archivo db.php
require_once 'db.php';

// Si fue enviado el formulario se procesará
if (isset($_POST['entrar']) && $_POST['entrar'] == 'Entrar' && isset($_POST['token']) &&
$_POST['token'] == $_SESSION['token']) {
        // Se verifican los datos de login
        if (isset($_POST['username'])) {
                $username = htmlentities(trim($_POST['username']));
        }
        if (isset($_POST['password'])) {
                $password = htmlentities(trim($_POST['password']));
        }

        $consulta = "SELECT * FROM Usuarios WHERE username =
'".mysql_escape_string($username)."' AND password = '".mysql_escape_string($password)."'";
        $resultado = mysql_query($consulta);
        // Se evalua si el usuario existe y se encuentra habilitado
        if (mysql_num_rows($resultado) > 0) {
                // Registrar variables de sesión
                $_SESSION['usuarioRegistrado'] = true;
                $_SESSION['username'] = $username;
                $_SESSION['http_user_agent'] = md5($_SERVER['HTTP_USER_AGENT']);
                // Se regenera el id de la sesión
                session_regenerate_id();
                // Se direcciona a la página de aplicaciones
                header("Location: loginok.php");
        } else {
                // Se hace una redirección por un mal login
                header("Location: loginfail.php");
        }
}
```

chequear.php
```
<?php
session_start();

if ($_SESSION['http_user_agent'] != md5($_SERVER['HTTP_USER_AGENT'])) {
        header("Location: login.php");
}
```

loginok.php
```
<?php
require_once 'chequear.php';

echo "Login Ok!!!";
echo "<br/>";
echo "Usuario: ".htmlentities($_SESSION['username']);
```

3) – Modifique la opción que permita reconocer las etiquetas reducidas en PHP. Luego escriba un script (phpinfo.php) para probar el reconocimiento.

php.ini
```
short_open_tag = On
```

phpinfo.php
```
<?
        phpinfo();
?>
```

4) – Modifique la opción de php.ini que permita desactivar el uso de la función phpinfo() en el servidor Web que utiliza PHP. Pruebe ejecutar el script phpinfo.php que ver los resultados.

php.ini
```
disable_functions = phpinfo
```

phpinfo.php
```
<?
        phpinfo();
?>
```

Anotaciones de los Ejercicios