Trabajo Práctico N° 6

En la sexta entrega del trabajo práctico continuaremos modificando nuestro sistema de gestión de usuarios aplicando los conocimientos adquiridos sobre las distintas técnicas de seguridad para evitar ataques en nuestra aplicación. Para ello partiremos de un sistema base el cual deberemos ir completando ya sea creando clases o editando los scripts ya desarrollados.

Objetivo: Comprender y profundizar las técnicas de seguridad y evitar posibles ataques contra nuestras aplicaciones y de esta manera poder brindar a nuestros usuarios sitios confiables y seguros.

Introducción: En el mundo de la informática existe una gran variedad de ataques cuyo objetivo es comprometer nuestros equipos y sistemas y de esta manera engañar a los usuarios para obtener información o dañar su patrimonio. En el ambiente de la programación existen diversas técnicas para evitar estos ataque y de estar forma poder proteger a nuestros usuarios garantizando su seguridad y la de sus datos.

Puntos a realizar en la entrega:

1. Filtrado de datos de entrada y salida.
2. Prevención de ataques de SQL Injection.
3. Prevención de falsificación de formularios y ataques de CSRF.
4. Prevención contra ataques de Fijación de Sesión y Session Hijacking.

Desarrollo

1. Filtrado de datos de entrada y salida.

Utilizando el conjunto de funciones ctype_, verificar que los parámetros recibidos desde los formularios (alta, editar, etc.) sean del tipo esperado.

Utilizando la función htmlentities o htmlspecialchars filtrar los parámetros necesarios recibidos desde los formularios para evitar ataques de XSS.

2. Prevención de ataques de SQL Injection.

Utilizando las funcionalidades de queries preparadas de PDO, mitigue todos los posibles ataques de SQL Injection en todas las sentencias SQL las cuales reciban valores para su definición.

3. Prevención de falsificación de formulario y ataques contra de CSRF.

En los formularios de la aplicación, aplicar la técnica en que se implementa un token guardado en sesión y el envío del mismo.

4. Prevención contra ataques de Fijación de Sesión y Session Hijacking.

Utilizar la función session_regenerate_id() en diferentes puntos de la aplicación para mitigar ataques de fijación de sesión.

En el proceso de login guardar en una variable de sesión alguna información que identifique al la unicidad del mismo (ej: El string de USER_AGENT que contiene el navegador), luego al chequear la información de la sesión, validar este dato y así evitando robo de sesiones.

Entrega

La sexta entrega deberá contener el proyecto con todos los scripts modificados, y deberá ser cargado a la plataforma como un único archivo comprimido en formato *.rar o *.tar.gz.