Reporte de Vulnerabilidad

Archivo: edit.php

Código Analizado:

```
<?php
include "models/conexion.php";
$id = $_GET["id"];
echo $id;

$sql = $conexion->query("SELECT * FROM PERSONAS WHERE id = $id");
?>
<!doctype html>
<html lang="en">

<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.5.2/css/all.min.css">


    <title>Hello, world!</title>
</head>

<body>

    <div class="container-fluid-row ">

        <div class="row px-4">
            <h1>Prueba Tecnica</h1>
            <div class="col">
                <div class="card">
                    <div class="card-body">
                        <form action="" method="post">
                            <!-- Mostrar Error     -->
                            <?php
                            include "controller/modificar.php";
                            // Suponiendo que la consulta ya fue ejecutada y $sql contiene un
solo registro
                            if ($a = $sql->fetch_object()) { ?>
                                <h3>Actudalizar Registro de Persona</h3>
                                <input type="hidden" name="id" value="<?= htmlspecialchars($a-
>id) ?>">
                                <div class="mb-3">
                                    <label for="nombre" class="form-label">Nombre</label>
                                    <input type="text" class="form-control" id="nombre"
name="nombre" value="<?= htmlspecialchars($a->nombre) ?>" required>
                                </div>
                                <div class="mb-3">
                                    <label for="apellido" class="form-label">Apellido</label>
                                    <input type="text" class="form-control" id="apellido"
name="apellido" value="<?= htmlspecialchars($a->apellido) ?>" required>
                                </div>
                                <div class="mb-3">
                                    <label for="dni" class="form-label">Cedula</label>
                                    <input type="text" class="form-control" id="dni"
name="dni" value="<?= htmlspecialchars($a->cedula) ?>" required>
                                </div>
                                <div class="mb-3">
                                    <label for="email" class="form-label">Email</label>
                                    <input type="email" class="form-control" id="email"
name="email" value="<?= htmlspecialchars($a->correo) ?>" required>
                                </div>
                                <button type="submit" name="btnactualizar" class="btn btn-
warning" value="ok">Actualizar</button>
                                <a href="index.php" class="btn btn-secondary">Cancelar</a>
                            <?php } else { ?>
```

```
                                <p>No se encontró el registro</p>
                            <?php } ?>

                        </form>
                    </div>
                </div>
            </div>

        </div>
    </div>

    <!-- Optional JavaScript; choose one of the two! -->

    <!-- Option 1: Bootstrap Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>

    <!-- Option 2: Separate Popper and Bootstrap JS -->
    <!--
    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"
integrity="sha384-IQsoLXl5PILFhosVNubq5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
integrity="sha384-cVKIPhGWiC2Al4u+LWgxfKTRIcfu0JTxR+EQDz/bgldoEyl4H0zUF0QKbrJ0EcQF"
crossorigin="anonymous"></script>
    -->
</body>

</html>
```

Análisis: ```html
Vulnerabilidades de Seguridad

Inyección SQL

Línea Aproximada: `$sql = $conexion->query("SELECT * FROM PERSONAS WHERE id = $id");`

Descripción: La variable `$id` se obtiene directamente de `$_GET["id"]` y se utiliza en la consulta SQL sin ninguna validación o sanitización. Esto permite a un atacante inyectar código SQL malicioso manipulando el valor del parámetro `id` en la URL. Por ejemplo, un atacante podría usar `id=1 OR 1=1` para obtener todos los registros de la tabla `PERSONAS`.

Mitigación: Utilizar consultas preparadas (prepared statements) con parámetros enlazados (bound parameters). Esto asegura que los datos proporcionados por el usuario se traten como datos y no como parte del código SQL.

Métricas de Calidad del Código

Ausencia de Validación de Datos

Descripción: Los datos que se reciben del formulario (nombre, apellido, dni, email) no se validan ni se sanitizan antes de ser enviados al controlador `modificar.php`. Esto puede llevar a errores de procesamiento, inconsistencia en los datos y posibles problemas de seguridad (como XSS, si los datos se vuelven a mostrar en la página sin codificación).

Mejora: Implementar validación del lado del servidor para todos los campos del formulario. Esto puede incluir la comprobación de tipos de datos, formatos (por ejemplo, formato de correo electrónico), longitudes máximas y rangos válidos. Utilizar funciones de sanitización (como `htmlspecialchars()` para la salida y funciones de filtro para la entrada) para prevenir inyección de código.

Legibilidad

Descripción: Aunque el código es relativamente sencillo, la combinación de lógica PHP y HTML puede dificultar la lectura.

Mejora: Separar la lógica de presentación de la lógica de negocio tanto como sea posible. Considerar el uso de un motor de plantillas (como Twig o Blade) para facilitar la generación de HTML y mejorar la legibilidad. Comentar el código para explicar la función de las diferentes secciones.

Acoplamiento

Descripción: El código presenta un acoplamiento entre la capa de presentación (HTML) y la capa de acceso a datos (consulta SQL directamente en el script).

Mejora: Implementar un patrón de diseño como MVC (Modelo-Vista-Controlador) para separar la lógica de la aplicación en componentes distintos. Esto reducirá el acoplamiento y facilitará el mantenimiento y la escalabilidad del código. La consulta a la base de datos debería estar en el modelo y la presentación en la vista.

Solución Propuesta

El siguiente código demuestra cómo mitigar la vulnerabilidad de inyección SQL utilizando consultas preparadas:

```
<?php
include "models/conexion.php";

// Obtener el ID de la petición GET
$id = $_GET["id"];

// Preparar la consulta SQL
$sql = $conexion->prepare("SELECT * FROM PERSONAS WHERE id = ?");

// Vincular el parámetro
$sql->bind_param("i", $id); // "i" indica que $id es un entero

// Ejecutar la consulta
$sql->execute();

// Obtener el resultado
$result = $sql->get_result();
?>
<!doctype html>
<html lang="en">
<head>
    <!-- ... (resto del código HTML) ... -->
</head>
<body>

    <div class="container-fluid-row ">
        <div class="row px-4">
            <h1>Prueba Tecnica</h1>
            <div class="col">
                <div class="card">
                    <div class="card-body">
                        <form action="" method="post">
                            <!-- Mostrar Error     -->
                            <?php
                            include "controller/modificar.php";
                            // Suponiendo que la consulta ya fue ejecutada y $sql contiene un
solo registro
                            if ($a = $result->fetch_object()) { ?>
                                <h3>Actudalizar Registro de Persona</h3>
                                <input type="hidden" name="id" value="<?= htmlspecialchars($a-
>id) ?>">
                                <div class="mb-3">
                                    <label for="nombre" class="form-label">Nombre</label>
                                    <input type="text" class="form-control" id="nombre"
name="nombre" value="<?= htmlspecialchars($a->nombre) ?>" required>
                                </div>
                                <div class="mb-3">
                                    <label for="apellido" class="form-label">Apellido</label>
                                    <input type="text" class="form-control" id="apellido"
name="apellido" value="<?= htmlspecialchars($a->apellido) ?>" required>
                                </div>
                                <div class="mb-3">
                                    <label for="dni" class="form-label">Cedula</label>
                                    <input type="text" class="form-control" id="dni"
```

```
name="dni" value="<?= htmlspecialchars($a->cedula) ?>" required>
                                </div>
                                <div class="mb-3">
                                    <label for="email" class="form-label">Email</label>
                                    <input type="email" class="form-control" id="email"
name="email" value="<?= htmlspecialchars($a->correo) ?>" required>
                                </div>
                                <button type="submit" name="btnactualizar" class="btn btn-
warning" value="ok">Actualizar</button>
                                <a href="index.php" class="btn btn-secondary">Cancelar</a>
                            <?php } else { ?>
                                <p>No se encontró el registro</p>
                            <?php } ?>

                        </form>
                    </div>
                </div>
            </div>

        </div>
    </div>

    <!-- Optional JavaScript; choose one of the two! -->

    <!-- Option 1: Bootstrap Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>

    <!-- Option 2: Separate Popper and Bootstrap JS -->
    <!--
    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.2/dist/umd/popper.min.js"
integrity="sha384-IQsoLXl5PILFhosVNubqL5LC7Qb9DXgDA9i+tQ8Zj3iwWAwPtgFTxbJ8NT4GN1R8p"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.min.js"
integrity="sha384-cVKIPhGWiC2Al4u+LWgxfKTRIcfu0JTxR+EQDz/bgldoEyl4H0zUF0QKbrJ0EcQF"
crossorigin="anonymous"></script>
    -->
</body>
</html>
```

Explicación:

- `$conexion->prepare("SELECT * FROM PERSONAS WHERE id = ?")`: Prepara la consulta SQL con un marcador de posición (`?`) para el valor de `id`.
- `$sql->bind_param("i", $id)`: Enlaza el valor de `$id` al marcador de posición. El primer argumento (`"i"`) especifica el tipo de dato del parámetro (en este caso, un entero). Esto asegura que `$id` se trate como un valor y no como parte del código SQL.
- `$sql->execute()`: Ejecuta la consulta preparada.
- `$sql->get_result()`: obtiene el resultado de la consulta.

Otras Mejoras:

- **Validación de Entrada:** Asegurarse de que `$_GET["id"]` es un entero antes de usarlo en la consulta. Esto puede hacerse utilizando `filter_var($_GET["id"], FILTER_VALIDATE_INT)`.
- **Manejo de Errores:** Agregar manejo de errores para las operaciones de la base de datos. Por ejemplo, verificar si la consulta se ejecutó correctamente y mostrar un mensaje de error si no.

```
```