Riesgos de No Sanitizar Datos en PHP

✨Objetivo

Mostrar ejemplos reales de qué puede pasar si no se sanitizan correctamente los datos de entrada en una aplicación PHP.

🌙1. Ejecución de HTML (XSS)

Código vulnerable:

```
$nombre = $_GET['nombre'];
echo "Hola, $nombre!";
```

URL maliciosa:

```
http://tusitio.com/index.php?nombre=<script>alert('Hackeado')</script>
```

Resultado:

Se ejecuta JavaScript en el navegador del usuario. Puede robar cookies o mostrar contenido engañoso.

Solución segura:

```
$nombre = htmlspecialchars($_GET['nombre'], ENT_QUOTES, 'UTF-8');
echo "Hola, $nombre!";
```

🌿2. Inyección SQL (SQL Injection)

Código vulnerable:

```
$usuario = $_GET['usuario'];
$consulta = "SELECT * FROM usuarios WHERE nombre = '$usuario'";
```

Entrada maliciosa:

```
usuario=' OR 1=1 --
```

Resultado:

Consulta resultante:

```
SELECT * FROM usuarios WHERE nombre = '' OR 1=1 --';
```

El atacante accede a todos los registros.

Solución segura (PDO):

```
$stmt = $pdo->prepare("SELECT * FROM usuarios WHERE nombre = :nombre");
$stmt->execute(['nombre' => $_GET['usuario']]);
```

🕳️3. Inyección en cabeceras HTTP

Código vulnerable:

```
header("Location: pagina.php?usuario=" . $_GET['usuario']);
```

Entrada maliciosa:

```
usuario=Juan%0d%0aSet-Cookie:admin=true
```

Resultado:

El atacante podría inyectar cabeceras y modificar el comportamiento de la respuesta HTTP.

🐞4. Archivos maliciosos

Carga permitida sin validación:

El atacante sube un archivo `inofensivo.php` con este contenido:

```
<?php system($_GET['cmd']); ?>
```

Luego accede a:

```
http://tusitio.com/uploads/inofensivo.php?cmd=rm -rf /
```

Resultado:

Ejecución remota de comandos en el servidor.

Solución:

- Validar extensión y tipo MIME.
- Guardar archivos con nombres aleatorios.
- Nunca ejecutar directamente archivos subidos.

⚡Resumen Comparativo

Situación	Entrada maliciosa	Resultado
XSS en nombre	`<script>`	Ejecuta JS en navegador
SQL Injection	`' OR 1=1 --`	Acceso no autorizado a la BD
Cabecera HTTP	`%0d%0a...`	Inyección de cookies u otras cabeceras
Archivo malicioso	`.php` con código	Ejecución remota de comandos

---

🐟Actividad sugerida

Crear un formulario de contacto vulnerable y corregirlo aplicando funciones de sanitización como `htmlspecialchars()`, `filter_var()`, y el uso de PDO.

🌧️Conclusión

- No confiar nunca en los datos del usuario.
- Validar y sanitizar siempre.
- Aplicar buenas prácticas y funciones seguras.
- Mejor prevenir que corregir luego de un incidente.