

Alain's Phone



BILBOKO
INGENIARITZA
ESKOLA
ESCUELA
DE INGENIERÍA
DE BILBAO

**Alain Pedrueza, Unai De Leon, Oscar
Basaguren, Adrian Fernandez y Aitor Gonzalo**

25/11/2023
UPV-EHU



Alain's Phone

ÍNDICE

1.Link del repositorio

2.INTRODUCCIÓN

3.VULNERABILIDADES A SOLUCIONAR

3.1.● Rotura de control de acceso

3.2.● Fallos criptográficos

3.3.● Inyección

3.4.● Diseño inseguro

3.5.● Configuración de seguridad insuficiente

3.6.● Componentes vulnerables y obsoletos

3.7.● Fallos de identificación y autenticación

3.8.● Fallos en la integridad de datos y software

3.9.● Fallos en la monitorización de la seguridad

3.10.● Otras vulnerabilidades

4.BIBLIOGRAFÍA

Link del repositorio

<https://github.com/adrianfd3z/AlainPhone>

INTRODUCCIÓN

Tras haber realizado la primera práctica que consistía en el desarrollo de una página web sin tener en cuenta los aspectos de seguridad, en esta segunda entrega debemos centrarnos en mejorar la seguridad de nuestra página para evitar el mayor número de vulnerabilidades posibles.

Para ello, utilizaremos la herramienta ZAP, es una aplicación que nos permite realizar pentesting, nos aporta un informe de vulnerabilidad que utilizaremos como recurso para encontrar soluciones a los fallos que han sido detectados. La solución de los fallos es muy importante ya que, de acuerdo con los resultados de la encuesta realizada por ESET Latinoamérica para el documento ESET Security Report 2014, la explotación de vulnerabilidades se ha convertido en la mayor preocupación de las empresas en materia de seguridad, seguida de otros incidentes como Infección por malware, fraudes, phishing o ataques de denegación de servicio (DoS).

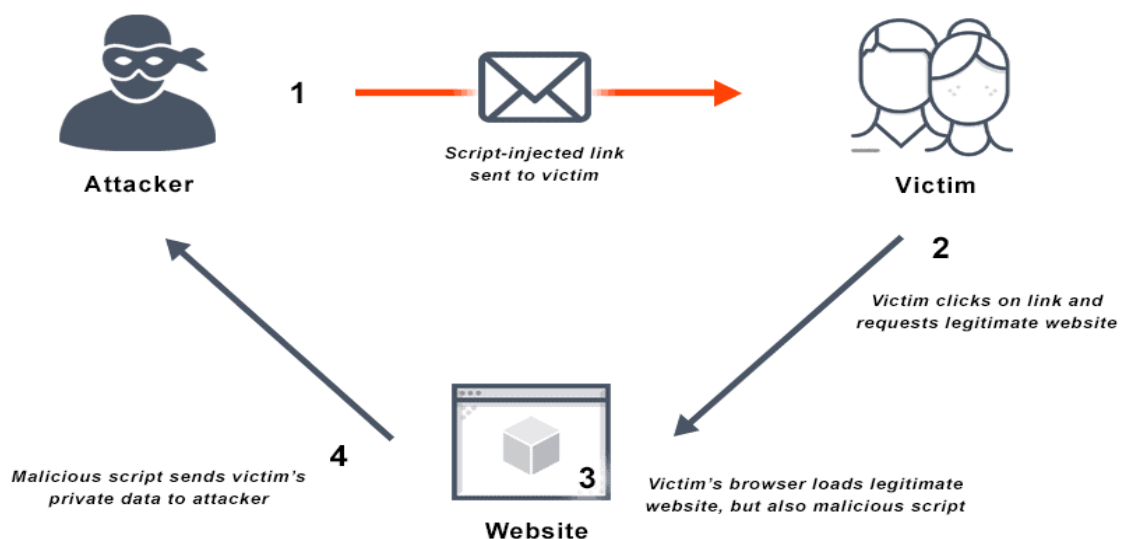
Como objetivo de esta práctica, se ha realizado una auditoría para evaluar la carencia de seguridad de nuestra página web y por tanto realizar un aumento de esta, aportando soluciones a las vulnerabilidades más comunes que la página presentaba.

VULNERABILIDADES A SOLUCIONAR

- **Rotura de control de acceso**

La "rotura de control de acceso" generalmente se refiere a una violación o fallo en el sistema de control de acceso de un lugar o sistema informático. El control de acceso es una medida de seguridad que regula quién tiene permiso para acceder a determinados recursos o áreas. Una rotura en este control significa que alguien ha logrado eludir o superar las restricciones establecidas.

Una de las técnicas más famosas es el Cross Site Scripting. Es una técnica de ataque que comprende hacer eco del código que fue proporcionado por el atacante en la instancia del navegador de un usuario.



Una de las opciones para prevenir este tipo de ataques es utilizar la validación de los campos de un formulario. Si el script PHP espera un integer de un input, cualquier otro tipo de dato debe rechazarse.

Cada dato debe ser validado cuando se recibe para asegurarse que es del tipo correcto, y rechazado si no pasa ese proceso de validación. En nuestro caso hemos hecho uso de expresiones regulares para validar los tipos de datos introducidos en los campos de los formularios. Por ejemplo:

```
if(!([a-zA-Z]+$/gm.test(document.getElementById('nombre').value))){  
//Comprueba si
```

```

el campo es vacío o contiene números ()
alert("El campo nombre está vacío o contiene un número");
return false;
}

```

Con este if comprobamos que el nombre a la hora de hacer el registro sólo contiene letras, es decir no puede ser ni vacío, ni contener números ni ningún tipo de carácter especial.

También nos debemos asegurar de que no se puedan acceder a las funciones que requieren registro sin haber iniciado sesión previamente. Para ello comprobaremos que la sesión del usuario existe:

```

(if(!isset($_SESSION['Usuario']) || !isset($_SESSION['DNI']))){
header("location:iniciosesion.php");
}
else{ .....
}}

```

Si existe, el usuario podrá acceder a la página sin ningún problema pero en el caso de que no exista, se le redirigirá automáticamente al formulario de inicio de sesión para que introduzca sus credenciales. Por ejemplo, si en la barra del navegador introducimos el `http://localhost:81/areapersonal.php` y no hay ningún usuario con la sesión iniciada, no se podrá acceder y por lo tanto se redirigirá a `http://localhost:81/iniciosesion.php`.

Además en el caso de que un usuario inicie sesión y esté más de una hora con la sesión iniciada, se cerrará su sesión y se le redirigirá al formulario de inicio sesión.

```

$inactivo=3600;
if(isset($_SESSION['tiempo']) ) {
$vida_session = time() - $_SESSION['tiempo'];
if($vida_session > $inactivo){
session_unset();
session_destroy();
echo '<script language="javascript">alert("Se ha caducado tu sesión!Vuelve a iniciarsesión");
window.location="iniciosesion.php";</script>';
}
}

```

```

$dni=$_SESSION['DNI'];
if($datosusuario=$conectar->prepare("SELECT * FROM Usuario
WHERE(DNI=?)")){
$datosusuario->bind_param('s',$dni);

```

```

$datosusuario->execute();
$lista=$datosusuario->get_result();
$datosusuario->close();
$conectar->close();
}
}

```

```

$_SESSION['tiempo'] = time();
?>

```

● Fallos criptográficos

Los fallos criptográficos son debilidades o errores en la implementación o uso de técnicas criptográficas, que pueden comprometer la seguridad de los sistemas que dependen de la criptografía para proteger la información.

Existen diferentes fallos criptográficos que facilitan a una persona realizar un ataque criptográfico, estos ataques resultan en una exposición de los datos privados, en nuestro caso las contraseñas de los usuarios.

Ejemplos de estos fallos serían los siguientes:

-Si para encriptar unos datos se usa la función por defecto de la base de datos de encriptación, si hubiese un ataque de inyección SQL, cualquiera podría desencriptar los datos solo con consultarlos.

-La no obligatoriedad del uso de TLS hace que un usuario pueda pasar de conexión HTTPS a HTTP y en consecuencia no asegurar la seguridad que proporciona el HTTPS.

Para solucionar este problema vamos a almacenar el hash de las contraseñas de los usuarios.

Por ejemplo en el inicio de sesión:

```

if(mysqli_num_rows($datosUsuario)>0){
$hash=mysqli_fetch_array($contrasenaAcomparar)[0];

```

Este sería el hash dentro de la base de datos

```

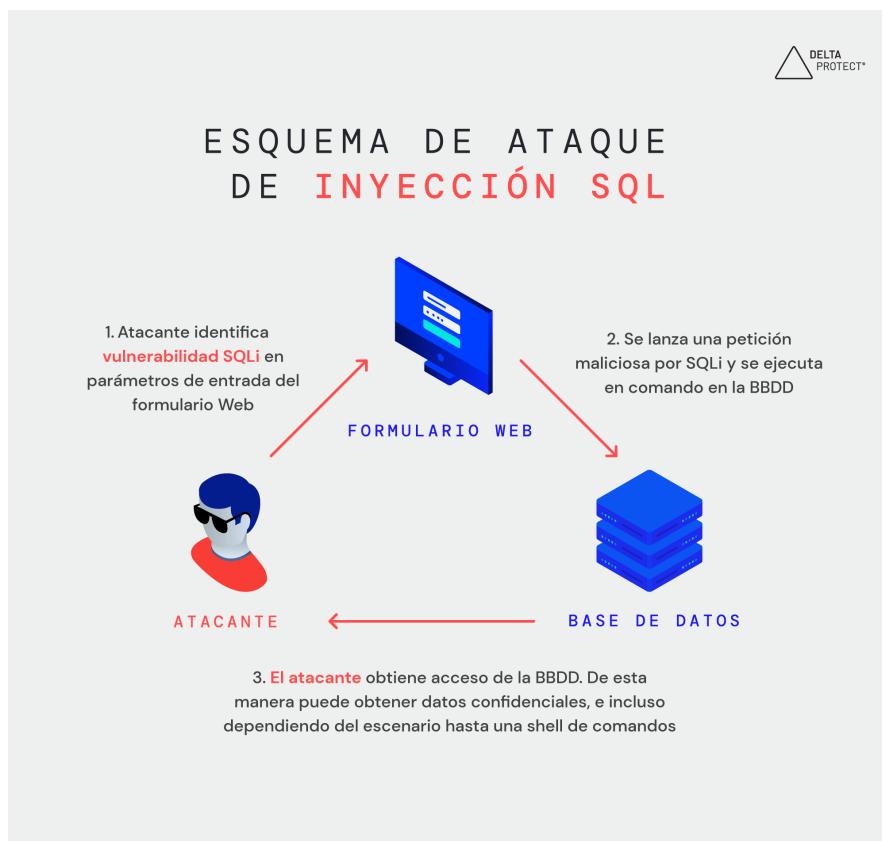
if(password_verify($contrasena,$hash)){

```

Y con esta función comprueba que el hash de la contraseña introducida sea el mismo

• Inyección

Las inyecciones de SQL (o SQLi) se producen cuando el hacker introduce o inyecta en el sitio web código SQL malicioso, un tipo de malware que se conoce como la carga útil, y consigue de manera disimulada que envíe ese código a su base de datos como si de una consulta legítima se tratara.



Para solucionar este problema haremos uso de consultas SQL parametrizadas. Las consultas parametrizadas utilizan una serie de parámetros que son unos valores que se añaden a la consulta al ejecutarla y son analizadas de forma literal. Sus ventajas son múltiples, por ejemplo: dificulta el acceso a los datos guardados en la base de datos, se reduce el tiempo de una consulta y los parámetros enlazados minimizan el ancho de banda consumido del servidor.

Cómo se ejecutaba una instrucción antes:

```
$sql1="UPDATE Movil SET Modelo ='$modelo'WHERE (Modelo='$Modelo' AND SistemaOperativo='$sistema_operativo' AND Gama='$Gama')";
```

```
$ejecutar1=mysqli_query($conectar,$sql1);
```

Cómo se ejecuta una instrucción ahora:

```

if($sql = $conectar->prepare("UPDATE Movil SET Modelo=?, Marca=?,
Precio=?, Gama=?, SistemaOperativo=? WHERE Modelo=? AND
SistemaOperativo=? AND Gama=? AND DNIDueño=?")){
    $sql->bind_param('ssissssss', $modelo, $marca, $precio, $gama,
    $sistop, $Modelo, $sistema_operativo, $Gama, $dniDueño);
    $sql->execute();
    $ejecutar=$sql->get_result();
    if(!$ejecutar){
        ?>
        <h3 class="bien">¡Datos modificados correctamente!</h3>
        <?php
    }
    $sql->close();
    header("Location:listapersonal.php");
}

```

El comando prepare crea una plantilla de la sentencia SQL se crea y se envía a la base de datos. Algunos valores se dejan sin especificar, llamados parámetros y representados por un interrogante "?"

Luego se les asigna un valor a los parámetros con el comando bind_param, donde la "s" significa que es un string, la "i" significa que es un integer, la "d" significa que es double y la "b" significa que es un blob (objeto binario grande).

- **Diseño inseguro**

Los errores en el apartado de diseño inseguro, no tienen que ver tanto con el código si no con la arquitectura de la página web. Esto quiere decir que existen opciones que no deberían estar disponibles para todos los usuarios, por ejemplo en nuestro caso el modo editar y eliminar móviles o ver información sensible de otros usuarios. Para ello se han realizado varios cambios para asegurarse que ninguna persona no registrada pueda tener el mismo rol que una que se haya registrado o un administrador

Ejemplo, pestaña **Ver Móviles**:

Antes:

Móviles registrados						
Modelo	Marca	Precio	Gama	Sistema Operativo	Editar	Eliminar
iPhone15	Apple	1000	Alta	IOS	Editar	Eliminar
iPhone8	Apple	400	Media	IOS	Editar	Eliminar
Galaxy Mini	Samsung	40	Baja	Android	Editar	Eliminar

Ahora:

móviles registrados con el DNI: 16094719D						
Modelo	Marca	Precio	Gama	Sistema Operativo	DNI Dueño	
a	rlp	123	Alta	IOS	16094719D	Editar Eliminar
AdrianMóvil	AlainPhone	200000	Altbaja	ALAIN	16094719D	Editar Eliminar

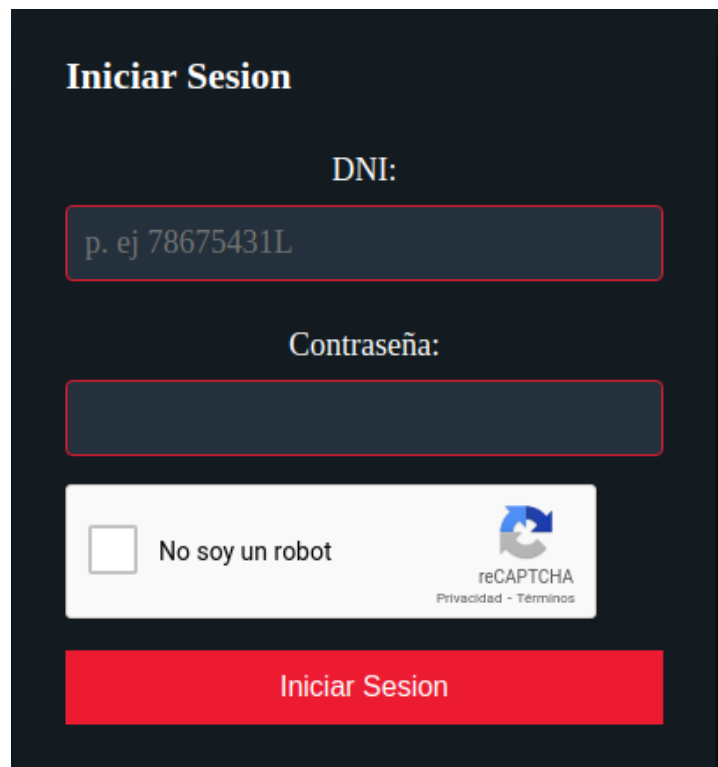
Se ha eliminado la opción de editar y eliminar móviles, ya que cualquier usuario registrado o no podía modificar los datos de los móviles registrados en la base de datos o incluso introducir errores intencionados o eliminarlos de ella, por ello se ha optado por rediseñar la interfaz eliminando las dos opciones.

Ahora a la hora de añadir un nuevo móvil, la opción está únicamente disponible para los usuarios registrados en el sistema.

También hemos implementado un CAPTCHA para evitar que robots/bots intenten hacer uso de nuestra página web.

Un CAPTCHA ayuda a proteger a la página web del spam y del descifrado de contraseñas pidiendo al usuario que complete una simple prueba que demuestre que es humano y no un ordenador que intenta acceder a la web. En nuestro caso hemos utilizado reCAPTCHA (reCAPTCHA v2) que es el sistema

Captcha que utiliza Google para detectar el tráfico procedente de programas automatizados o bots.



The image shows a login interface on a dark background. At the top, the text "Iniciar Sesion" is displayed in a light blue font. Below this, the label "DNI:" is followed by a text input field containing the placeholder text "p. ej 78675431L". Underneath the DNI field is the label "Contraseña:" followed by a password input field. Below the password field is a reCAPTCHA widget. The widget includes a checkbox with the text "No soy un robot" and the reCAPTCHA logo, which consists of a blue circular arrow and the text "reCAPTCHA" and "Privacidad - Términos". At the bottom of the form is a large red button with the text "Iniciar Sesion" in white.

Cuando pulsamos en “No soy un robot”, el sistema revisa el comportamiento que el usuario ha tenido anteriormente y en función de todo ello decide si el usuario es un usuario humano o un bot y en función de ello permite o prohíbe el acceso. Para analizar el comportamiento el sistema usa un algoritmo secreto. Este algoritmo usa, por ejemplo, las cookies activas y dirección IP para verificar el comportamiento a través de Internet. También analiza el movimiento de nuestro ratón para ver cómo nos comportamos para pulsar en el cuadro de la caja de reCAPTCHA. Los bots suelen hacerlo de una manera automática, mientras que los humanos no solemos ir siempre derechos a la caja seleccionable, y por lo que el recorrido de nuestro ratón es diferente. En el caso de que el comportamiento del usuario le haga dudar al sistema de tu naturaleza humana, el reCaptcha mostrará una ventana en la que te pedirá que escribas un texto o que hagas click sobre determinadas imágenes.

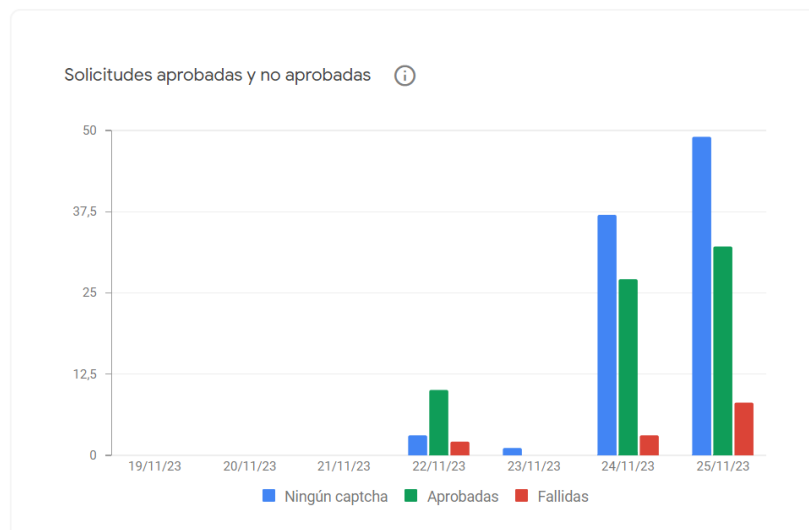
Por último, esta herramienta nos permite generar un informe con el número de solicitudes recibidas y el % de solicitudes sospechosas y seleccionar el nivel de dificultad y la opción de enviar alertas al administrador(nosotros) si aumenta el tráfico sospechoso en la web (muchas peticiones en muy poco tiempo).

Total de verificaciones

154

Solicitudes sospechosas

0 %



Informe de las solicitudes reCAPTCHA v2 en los formularios

Preferencia de seguridad



La más fácil para los usuarios

La más segura

☒ Verificar el origen de las soluciones de reCAPTCHA

Si se inhabilita, debes [comprobar el nombre de host](#) en tu servidor cuando verifiques una solución.

☐ Enviar alertas a los propietarios

Recibe alertas si Google detecta algún problema en tu sitio web, como errores de configuración o un aumento del tráfico sospechoso.

● Configuración de seguridad insuficiente

Esta vulnerabilidad es muy general y por tanto abarca fallos y soluciones ya explicadas en anteriores o posteriores apartados, por ejemplo: la gestión de errores que desvela información (Fallos en la monitorización de la seguridad), el uso de cuentas y contraseñas por defecto (Fallos de identificación y autenticación) o el problema de los bots en nuestro sistema (Diseño Inseguro). Como hemos visto este apartado incluye muchos de los apartados ya vistos y explicados, por lo tanto las distintas vulnerabilidades ya han sido tratadas.

● Componentes vulnerables y obsoletos

Es imprescindible saber la versión de los componentes que hemos utilizado en el desarrollo de la página web. Además, hay que destacar que se debe comprobar regularmente que el software utilizado no debe ser obsoleto ni vulnerable. Es importante escanear regularmente el sistema de forma que se puedan evitar vulnerabilidades.

Para solucionar este problema se ha optado por listar las versiones de los componentes que utilizamos y cambiar los “latest” por la última versión que se conoce, además de que en un futuro deberemos actualizar a la última versión estos componentes y desde la fuente oficial.

Versiones de los componentes:

phpMyAdmin: **5.2.1**

Mariadb:**10.11**

Antes en el docker-compose.yml:

```
phpmyadmin:
```

```
image: phpmyadmin/phpmyadmin:latest
```

Ahora:

```
phpmyadmin:
```

```
image: phpmyadmin/phpmyadmin:5.2.1
```

● Fallos de identificación y autenticación

Este es uno de los fallos de seguridad más comunes en los sistemas web. Es de vital importancia que se compruebe y se confirme la identidad del usuario debidamente y proteger sus datos como las contraseñas, nombres, información personal, etc. Es por eso, que hemos realizado algunos cambios respecto a la página web de la primera entrega, ya que era totalmente vulnerable en autenticación, puesto que se permitían el uso de contraseñas débiles, ataques de fuerza bruta, las contraseñas se almacenaban en texto plano en la BD, quedando así expuestas a un sencillo ataque , o no se invalidaba la sesión después de un periodo de actividad.

Todo esto se ha mitigado y corregido, como se expone a continuación:

Para solucionar este problema, hemos modificado el usuario y la contraseña de PHPMyAdmin ya que las credenciales anteriores (**usuario: admin y contraseña: test**) eran bastante sencillas y fáciles de adivinar.

Las nuevas credenciales no son fáciles de adivinar, con lo que la seguridad del gestor de la base de datos aumenta muchísimo. Según un comprobador de contraseñas de la prestigiosa empresa Kaspersky esta contraseña se descifraría en aproximadamente 3261 siglos.(Ilustración 8)

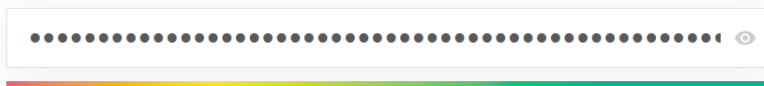
Además, se ha añadido a la hora de registrarse un el usuario no podrá usar contraseñas débiles o conocidas , y se comprobará que la contraseña cumple con los requisitos con la siguiente expresión regular:

```
if(!/(?=\w*\d)(?=\w*[A-Z])(?=\w*[a-z])\S{8,16}$/.test(document.getElementById('contrasena').value)){  
    alert("La contraseña no cumple los requisitos!");  
    return false;  
}
```

Por otro lado, se ha tomado la decisión de almacenar en la BD el hash de las contraseñas para que no sean fácilmente reconocibles si no se conoce el algoritmo utilizado dificultando severamente los ataques. Para esto, hemos optado por el algoritmo bcrypt, que es una función de hashing de passwords diseñado por Niels Provos y David Maxieres, basado en el cifrado de Blowfish.

```
$contrasena=password_hash($_POST['contrasena'],PASSWORD_DEFAULT."  
T."n");
```

Cabe destacar, que el fallo de invalidación de sesión tras un tiempo de inactividad se ha resuelto ya en apartado anterior de la documentación de la Rotura del control de acceso.

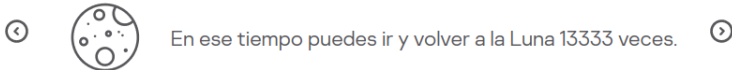


✓ ¡Buena contraseña!

- Tu contraseña es resistente al pirateo.
- Tu contraseña no aparece en ninguna base de datos de contraseñas filtradas.

Tu contraseña puede ser descifrada con un ordenador común en...

3261 siglos



En ese tiempo puedes ir y volver a la Luna 13333 veces.



[¿Quieres aprender cómo crear contraseñas súper fuertes? ¡Da clic aquí!](#)

Docker-compose.yml antes

```
db:
environment:
  MYSQL_ROOT_PASSWORD: root
  MYSQL_USER: admin
  MYSQL_PASSWORD: test
phpmyadmin:
environment:
  MYSQL_USER: admin
  MYSQL_PASSWORD: test
```

Docker-compose.yml ahora

```
db:
environment:
  MYSQL_ROOT_PASSWORD: yKbfwxEjRNByKz
  MYSQL_USER: DgfG5eVE1pEb8B
  MYSQL_PASSWORD: yKbfwxEjRNByKz
phpmyadmin:
```

environment:

MYSQL_USER: DgfG5eVE1pEb8B

MYSQL_PASSWORD: yKbfwxEjRNByKz

- **Fallos en la integridad y datos del software**

Este tipo de errores se relacionan con código e infraestructura no protegidos contra alteraciones (integridad). Esto ocurre por ejemplo cuando una aplicación depende de bibliotecas o plugins. Como nosotros no hemos utilizado ningún tipo de biblioteca o plugin no nos afecta este fallo, pero en caso de añadirlas deberíamos de tener en cuenta sus posibles consecuencias y asegurarnos de usar las versiones más recientes.

- **Fallos en la monitorización de la seguridad**

En nuestra antigua versión de la página web, al ocurrir un error, se mostraban información relevante, como la ubicación del error en los archivos. Para evitarlo hemos creado un registro de errores llamado erroresnuevo.log, para que esos errores en vez de mostrarse por pantalla se guarden en el log.

```
24/11/2023 12:36:04 Contraseña incorrecta,usuario con DNI 72231041R La ip del login es: 172.17.0.1
24/11/2023 12:36:45 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
24/11/2023 17:22:59 ¡Ha ocurrido un error en el registro,vuelve a introducir los datos! La ip del login es: 172.17.0.1
24/11/2023 17:26:40 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
24/11/2023 18:25:13 Contraseña incorrecta,usuario con DNI 72231041R La ip del login es: 172.17.0.1
24/11/2023 18:27:43 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
24/11/2023 19:06:15 Se ha registrado el usuario con DNI79224746G La ip del login es: 172.17.0.1
24/11/2023 19:11:19 Contraseña incorrecta,usuario con DNI 79224746G La ip del login es: 172.17.0.1
24/11/2023 19:12:20 Se ha registrado el usuario con DNI79224746G La ip del login es: 172.17.0.1
24/11/2023 19:33:16 Se ha registrado el usuario con DNI79224746G La ip del login es: 172.17.0.1
24/11/2023 19:34:03 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 11:07:14 Se ha registrado el usuario con DNI72264643T La ip del login es: 172.17.0.1
25/11/2023 12:10:23 El usuario no existe! DNI introducido: alain@gmail.com La ip del login es: 172.17.0.1
25/11/2023 12:10:37 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 12:13:04 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 12:33:43 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 13:44:33 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 13:45:05 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 14:08:31 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 14:10:47 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 14:10:53 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 14:11:56 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 14:17:47 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
25/11/2023 15:38:18 Se ha iniciado sesión correctamente La ip del login es: 172.17.0.1
```

Función para loguear:

```
<?php
function getRealIP() {
    if (!empty($_SERVER['HTTP_CLIENT_IP']))
        return $_SERVER['HTTP_CLIENT_IP'];

    if (!empty($_SERVER['HTTP_X_FORWARDED_FOR']))
        return $_SERVER['HTTP_X_FORWARDED_FOR'];

    return $_SERVER['REMOTE_ADDR'];
}

function logear_error(String $mensaje){
    $conectar = @mysqli_connect("db", "DgfG5eVE1pEb8B", "yKbfxwEjRNByKz",
"database");
    if(!$conectar){
        echo"No Se Pudo Conectar Con El Servidor";
    }else{
        $base=mysqli_select_db($conectar,"database");
        if(!$base){
            echo"No Se Encontro La Base De Datos";
        }
    }
    $new_ip=getRealIP();
    $logFile = fopen("erroresnuevo.log", 'a+b');
    fwrite($logFile, "\n".date("d/m/Y H:i:s")."  ".$mensaje."  ".$La ip del login es:
".".$new_ip);
    fclose($logFile);
    if($log=$conectar->prepare("INSERT INTO Log VALUES(?,?)")){
        $log->bind_param('ss',htmlspecialchars(mysqli_real_escape_string($conectar,$
mensaje)),htmlspecialchars(mysqli_real_escape_string($conectar,$new_ip)));
        $log->execute();
        $log->close();
    }
}
?>
```


Además, en la base de datos también se añaden en una nueva tabla llamada log.

```
CREATE TABLE `Log` (  
  `Descripcion` varchar(50) NOT NULL,  
  `IP` varchar(50) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

Descripcion	IP
Se ha registrado el usuario con DNI79224746G	172.17.0.1
Contraseña incorrecta,usuario con DNI 79224746G	172.17.0.1
Contraseña incorrecta,usuario con DNI 79224746G	172.17.0.1

- **Otras vulnerabilidades**

Clickjacking

El clickjacking es un tipo de ataque en línea donde los usuarios son engañados para hacer clic en elementos maliciosos en una página web sin su conocimiento. Los atacantes ocultan elementos sobre la página legítima, haciendo que los clics realicen acciones no deseadas, como compartir información o descargar malware. Para prevenirlo hemos utilizado la cabecera `header('X-Frame-Options:SAMEORIGIN')`.



Ataques MIME

Nuestra página era vulnerable a ataques basados en confusión de tipos MIME, para evitarlo hemos introducido la cabecera header ('X-Content-Type-Options: nosniff'). De esta manera evitamos que se carguen hojas de estilo o scripts maliciosos en nuestra web.

BIBLIOGRAFÍA:

Clickjacking:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options>

RegEx contraseña:

<http://w3.unpocodetodo.info/utiles/regex-ejemplos.php?type=psw>

Inyección:

<https://norfipc.com/inf/como-proteger-formularios-web-evitar-inyeccion-codigo-sql.php>

IP en los logs:

<https://ejemplocodigo.com/ejemplo-php-obtener-la-ip-real-de-una-visita/>

reCAPTCHA:

<https://www.youtube.com/watch?v=M1jkZx2crBg>

<https://www.google.com/recaptcha/about/>

Ataques MIME:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options>

Rotura de control de acceso:

<http://www.forosdelweb.com/f18/limitar-tiempo-las-sesiones-993596/>

Log de errores:

<https://www.php.net/manual/es/function.fopen>

<https://www.php.net/manual/es/function.fwrite>

ChatGPT:

<https://chat.openai.com/>