Evaluación de Seguridad

Sistema de Login Seguro

Materia: Desarrollo de Software Seguro

6to Semestre

Basado en ISO/IEC 27001

Contenido

1. Descripción del Sistema de Login

El sistema **LoginSeguro** es una aplicación web desarrollada en **PHP** con base de datos **MySQL**, que implementa funcionalidades de registro e inicio de sesión de usuarios con un enfoque en la seguridad.

Tecnologías utilizadas

Componente	Tecnología
Backend	PHP 8.x
Base de datos	MySQL (PDO con prepared statements)
Frontend	HTML5, CSS3
Servidor	XAMPP (Apache + MySQL)

Funcionalidades principales

- Registro de usuarios con validación de contraseñas seguras (mínimo 8 caracteres, mayúsculas, minúsculas, números y caracteres especiales).
- Inicio de sesión con autenticación basada en hash SHA-256 con sal única por usuario.
- Panel de administración exclusivo para usuarios con rol **admin**.
- Protección contra fuerza bruta con bloqueo temporal tras 5 intentos fallidos.
- Protección CSRF mediante tokens únicos por formulario.
- Headers de seguridad HTTP (CSP, X-Frame-Options, X-XSS-Protection, etc.).
- Cookies de sesión configuradas con flags **HttpOnly**, **SameSite=Strict** y **Secure**.

2. Matriz de Riesgos (ISO/IEC 27001)

R1	**Fuga de Información**	**Impacto:** Alto	**Prob:** Baja	**Medio**
	Medidas de mitigación: Se oculta el header X-Powered-By. Se envían headers Cache-Control: no-store y Pragma: no-cache. Los mensajes de error son genéricos. Se implementa Referrer-Policy: strict-origin-when-cross-origin.			
R2	**Comunicación No Cifrada**	**Impacto:** Alto	**Prob:** Media	**Alto**
	Medidas de mitigación: Cookies con flag Secure cuando HTTPS está disponible. En desarrollo se usa HTTP. Recomendación: configurar HTTPS obligatorio con certificado SSL/TLS en producción.			
R3	**Inyección SQL**	**Impacto:** Crítico	**Prob:** Muy Baja	**Bajo**
	Medidas de mitigación: PDO con prepared statements en todas las consultas. Parámetros vinculados con placeholders. Entradas sanitizadas con htmlspecialchars(), trim() y stripslashes().			
R4	**Ataques de Fuerza Bruta**	**Impacto:** Alto	**Prob:** Baja	**Medio**
	Medidas de mitigación: Bloqueo temporal tras 5 intentos fallidos (15 min). Registro de intentos por usuario e IP. Requisitos de complejidad de contraseñas (8+ caracteres variados).			
R5	**XSS (Cross-Site Scripting)**	**Impacto:** Alto	**Prob:** Baja	**Medio**
	Medidas de mitigación: Escape con htmlspecialchars(). Header X-XSS-Protection. Content Security Policy (CSP). X-Content-Type-Options: nosniff.			
R6	**CSRF (Cross-Site Request Forgery)**	**Impacto:** Alto	**Prob:** Baja	**Medio**
	Medidas de mitigación: Tokens CSRF con random_bytes(32). Validación con hash_equals(). Regeneración tras cada uso. Cookies SameSite=Strict. X-Frame-Options: DENY.			
R7	**Almacenamiento Inseguro de Contraseñas**	**Impacto:** Crítico	**Prob:** Muy Baja	**Bajo**
	Medidas de mitigación: Salt + Hash SHA-256. Sal criptográfica de 32 bytes por usuario generada con random_bytes(). Verificación con hash_equals() contra timing attacks.			

3. Explicación de los Riesgos Identificados

3.1 Fuga de Información

Definición: Ocurre cuando el sistema revela información sensible (tecnologías usadas, existencia de usuarios, rutas internas, errores detallados) que un atacante puede usar para planificar un ataque.

En nuestro sistema: Se mitiga con mensajes de error genéricos que no indican si el usuario existe o no ("Credenciales inválidas. Verifique su usuario y contraseña."), headers que ocultan la tecnología del servidor, y políticas de caché que evitan el almacenamiento de datos sensibles en el navegador.

3.2 Comunicación No Cifrada

Definición: Cuando los datos viajan entre el cliente y el servidor sin cifrado, un atacante puede interceptar la comunicación mediante ataques Man-in-the-Middle (MitM) y capturar credenciales en texto plano.

En nuestro sistema: Las cookies están configuradas para activar el flag Secure cuando HTTPS está disponible. En el entorno de desarrollo local (HTTP) esto no se aplica, pero en producción se debe configurar HTTPS con certificado SSL/TLS.

3.3 Inyección SQL

Definición: La inyección SQL es una vulnerabilidad que permite a un atacante insertar código SQL malicioso en los campos de entrada, manipulando las consultas a la base de datos para acceder, modificar o eliminar información.

En nuestro sistema: Todas las consultas utilizan PDO con prepared statements. Los parámetros se vinculan de forma segura, impidiendo la inyección.

3.4 Ataques de Fuerza Bruta

Definición: Consisten en probar sistemáticamente todas las combinaciones posibles de contraseñas hasta encontrar la correcta. También incluyen ataques de diccionario donde se prueban contraseñas comunes.

En nuestro sistema: Se bloquea la cuenta tras 5 intentos fallidos durante 15 minutos, registrando cada intento con usuario, IP y timestamp. Las contraseñas deben ser complejas (8+ caracteres con variedad de tipos).

3.5 XSS (Cross-Site Scripting)

Definición: Vulnerabilidad que permite a un atacante inyectar scripts maliciosos en páginas web vistas por otros usuarios, pudiendo robar cookies, sesiones o redirigir a sitios maliciosos.

En nuestro sistema: Se escapan todas las salidas al HTML con htmlspecialchars(), se implementa Content Security Policy, y se habilita la protección XSS nativa del navegador.

3.6 CSRF (Cross-Site Request Forgery)

Definición: Ataque que fuerza a un usuario autenticado a ejecutar acciones no deseadas en una aplicación web en la que está autenticado, mediante enlaces o formularios maliciosos en otros sitios.

En nuestro sistema: Se generan tokens CSRF criptográficamente seguros para cada formulario, se validan en el servidor antes de procesar la solicitud, y las cookies SameSite=Strict previenen el envío automático de cookies en peticiones cross-origin.

3.7 Almacenamiento Inseguro de Contraseñas

Definición: Almacenar contraseñas en texto plano, con cifrado reversible, o con algoritmos de hash débiles (MD5, SHA-1 sin sal), permitiendo que un atacante con acceso a la base de datos obtenga las contraseñas originales.

En nuestro sistema: Se usa SHA-256 con sal única por usuario. Cada usuario tiene una sal generada con random_bytes(32), lo que significa que incluso si dos usuarios tienen la misma contraseña, sus hashes serán completamente diferentes. La verificación utiliza hash_equals() para evitar ataques de timing.

Ejemplo de Código — Prepared Statements (Inyección SQL)

```
$sql = "SELECT salt, hash_contrasena, rol FROM usuarios
WHERE nombre_usuario = :usuario AND activo = 1";
$stmt = $db->prepare($sql);
$stmt->execute([':usuario' => $usuario]);
```

Los parámetros se vinculan con placeholders, impidiendo que un atacante inyecte código SQL a través de los campos de entrada.

4. Norma ISO/IEC 27001

¿Qué es la norma ISO/IEC 27001 y cuál es su objetivo?

La **ISO/IEC 27001** es un estándar internacional publicado por la Organización Internacional de Normalización (ISO) y la Comisión Electrotécnica Internacional (IEC). Su objetivo es proporcionar un marco para establecer, implementar, mantener y mejorar continuamente un **Sistema de Gestión de Seguridad de la Información (SGSI)**.

El objetivo principal es **proteger la confidencialidad, integridad y disponibilidad** de la información a través de un proceso de gestión de riesgos, asegurando que se implementen controles de seguridad adecuados y proporcionales a los riesgos identificados.

¿Qué es un SGSI?

Un **SGSI** es un conjunto de políticas, procedimientos, directrices y recursos asociados, gestionados colectivamente por una organización para proteger sus activos de información. Incluye:

- **Evaluación de riesgos:** Identificar amenazas y vulnerabilidades.
- **Tratamiento de riesgos:** Implementar controles para mitigar los riesgos.
- **Monitoreo continuo:** Revisar y mejorar constantemente los controles.
- **Documentación:** Mantener registros de políticas y evidencias de cumplimiento.

Aplicación en el Desarrollo de Software

Control	Nombre	Descripción
A.8.25	Desarrollo seguro	Validación de entradas, manejo seguro de errores y control de acceso.
A.8.8	Gestión de vulnerabilidades	Pruebas de penetración y análisis de código.
A.8.3	Control de acceso	Autenticación robusta, gestión de roles y principio de mínimo privilegio.
A.8.24	Cifrado	Datos en tránsito (HTTPS/TLS) y en reposo (hashing de contraseñas).
A.8.15	Registro y monitoreo	Eventos de seguridad para auditoría.

5. Recomendaciones para Mejorar la Seguridad

#	Recomendación	Prioridad	Estado Actual
1	Implementar HTTPS obligatorio con certificado SSL/TLS para cifrar toda la comunicación.	Alta	No implementado
2	Migrar a bcrypt o Argon2 para el hashing de contraseñas (algoritmos adaptativos más resistentes que SHA-256).	Media	Usa SHA-256 con sal
3	Implementar autenticación de dos factores (2FA) con códigos TOTP (Google Authenticator).	Media	No implementado
4	Agregar límite de tasa (rate limiting) a nivel de servidor para complementar el bloqueo de fuerza bruta.	Media	Solo bloqueo por intentos
5	Implementar registro de auditoría completo con logs de todas las acciones administrativas.	Baja	Parcialmente implementado
6	Agregar política de expiración de sesiones para cerrar sesiones inactivas automáticamente.	Media	No implementado
7	Implementar CAPTCHA en login y registro para dificultar ataques automatizados.	Baja	No implementado