Universidad Autónoma de Baja California
Ensenada, Baja California, México

Facultad de Ingeniería, Arquitectura y Diseño

Parcial 3	
Materia:	Seguridad de Software
Alumno:	Diego Francisco Merino Huizar (367169)
Docente:	Marcel Celaya
Fecha:	31/05/2024

Seguridad Web

Objetivo del Proyecto

El objetivo principal de este proyecto es educar y concienciar a los desarrolladores sobre las vulnerabilidades de seguridad comunes en las aplicaciones web y proporcionar soluciones prácticas para mitigarlas. Cada módulo incluye ejemplos prácticos, demostraciones de ataques, y estrategias de defensa.

1. Implementación y Análisis de RSA
El algoritmo RSA es una técnica de cifrado asimétrico ampliamente utilizada para proteger la comunicación en la web. Fue inventado por Ron Rivest, Adi Shamir y Leonard Adleman en 1977. Desarrollado en el Instituto de Tecnología de Massachusetts (MIT) y patentado en 1983. Es fundamental en la criptografía moderna.

Funcionamiento: RSA utiliza dos claves, una pública y otra privada. Los mensajes cifrados con la clave pública sólo pueden ser descifrados con la clave privada correspondiente y viceversa.

Análisis de Seguridad

- Medidas de Mitigación: Uso de claves largas (2048 bits o más) y prácticas de seguridad adicionales.
- Impacto de la Longitud de Clave: A mayor longitud de clave, mayor seguridad pero también mayor coste computacional.

2. Análisis de Seguridad

Descripción General: Objetivos del Análisis: Evaluar la robustez y las posibles vulnerabilidades de la aplicación web.

Evaluación de la Seguridad de la Aplicación:

- Metodología Utilizada: Utilización de OWASP (Open Web Application Security Project) como marco de referencia.
- Identificación de Amenazas:
    - DDoS (Denegación de Servicio): Evaluación de la capacidad de la aplicación para resistir ataques de denegación de servicio.
    - Phishing: Evaluación de medidas para prevenir y detectar ataques de phishing.
    - XSS y SQL Injection: Identificación de vulnerabilidades relacionadas con XSS y SQL Injection.

Medidas de Mitigación

- Políticas de Seguridad:
    - Autenticación y Autorización: Implementación de autenticación multifactor y control de acceso basado en roles.
    - Gestión de sesiones: Uso de cookies seguras y limitación de la duración de las sesiones.
    - Manejo de Errores: Configuración adecuada para evitar la exposición de información sensible.
- Buenas Prácticas:
    - Validación y Sanitización de Entradas: Validación tanto en el servidor como en el cliente.
    - Uso de HTTPS: Implementación de HTTPS para proteger la transmisión de datos.
    - Actualizaciones y Parches Regulares: Mantener la aplicación y sus dependencias de seguridad.

3. XSS (Cross-Site Scripting)

Tipos de XSS

- Reflejado (Non-Persistent): El script malicioso se refleja en la respuesta del servidor
- Almacenado (Persistent): El script malicioso se almacena en el servidor y se entrega a los usuarios.
- Basado en DOM: El script malicioso se ejecuta en el navegador manipulando el DOM.

Medidas de mitigación

- Validación y Sanitización de Datos
    - Validación entradas en el servidor y el cliente
    - Uso de librerías como DON Purify para sanitizacion
- Content Security Policy (CSP):

```
Unset
<meta http-equiv="Content-Security-Policy" content="default-src 'self';
script-src 'self' https://trustedscripts.example.com">
```

4. SQL Injections

Tipos de SQL Injections

- Basada en Consultas: Modificaciones en las tablas en una consulta de SQL.
- Basada en Errores: Explotación de mensajes, warning etc.
- Basada en tiempo: Retrasar la respuesta del servidor para inferir la información.

```
Unset
// Ejemplo el ataque aqui seria OR '1' = '1'
SELECT * FROM users WHERE username = 'admin' AND password = 'password';
```

Medidas de mitigación

- Validación en entradas
- Principio mínimo privilegio: Configurar privilegios mínimos para los usuarios en cada base de datos

5. Phishing y Defensa

Tipos de Phishing

- Spear phishing: Ataques que van dirigidos a individuos específicos.
- Whale phishing: Ataques dirigidos a individuos de alto perfil
- Clone Phishing: Clonar correos para enviar mensajes maliciosos

```
Unset
Aqui tenemos un ejemplo de como decimos que si el servidor detecta ese
metodo POST el username y la password la guarda en el archivo de
credenciales.txt

<?php
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = $_POST['password'];
    file_put_contents('credentials.txt', "$username:$password\n",
FILE_APPEND);
  }
?>
```

Análisis de Efectividad

- Uso de ingenieria social para ganar la confianza del usuario
- Diseño parecido a sitios legítimos que son famosos (Ejemplo facebook)
- Dominio similares a los legítimos.

Medidas de Defensa

- Ejemplos de materiales educativos, como infografías y videos, con ejemplos explicativos.
- Filtros de correos para detectar y bloquear correos phishing.
- Implementación de autenticación de multifactor.
- Uso de herramientas de seguridad como antivirus, malware etc.