Departamento de Ciencias de la Computación
Universidad de las Fuerzas Armadas - ESPE

Taller de Investigación

Tema: OWASP API Security Top 10 - 2023

Nombres:
Yeshua Amador Chiliquinga Amaya
Cesar Ignacio Loor Mercado
Jose Migel Sanmartin Galan

Carrera / Asignatura: Ingeniería de Software / Ingeniería de Seguridad de Software
NRC: 2540
Nombre del profesor: Walter Fuertes, PhD

Fecha de presentación: 2 de diciembre del 2024

Índice

Índice de figuras

Índice de cuadros

1 Introducción

1.1 Objetivo

1.2 Objetivos Específicos

-

2 OWASP API Security Top 10 - 2023

El Open Web Application Security Project (OWASP) es una organización sin fines de lucro que se dedica a mejorar la seguridad de las aplicaciones web y las API. Una de sus contribuciones más reconocidas es el "OWASP Top 10", una lista que identifica las diez vulnerabilidades de seguridad más críticas en aplicaciones web. Este informe se actualiza regularmente para reflejar las amenazas más relevantes.

En 2019, OWASP reconoció la creciente importancia de las API y desarrolló una lista específica de vulnerabilidades para este tipo de servicios. La versión 2023 del OWASP API Security Top 10 identifica las principales amenazas que enfrentan las API modernas.

2.1 Herramientas OWASP para Resolver Vulnerabilidades

OWASP ofrece diversas herramientas y recursos para mitigar estas vulnerabilidades:

- **OWASP Testing Guide:** Guía detallada para realizar pruebas de seguridad en aplicaciones web y API, que ayuda a identificar vulnerabilidades específicas.
- **OWASP Cheat Sheet Series:** Hojas informativas con recomendaciones prácticas y ejemplos de código para evitar vulnerabilidades comunes.
- **OWASP ZAP (Zed Attack Proxy):** Herramienta gratuita y de código abierto para pruebas de penetración y detección de vulnerabilidades en aplicaciones web y API.
- **OWASP Dependency-Check:** Analiza dependencias y bibliotecas para identificar componentes vulnerables y ayudar en la gestión de riesgos.

- **OWASP ModSecurity Core Rule Set:** Conjunto de reglas para firewalls de aplicaciones web que ayuda a bloquear ataques comunes y proteger las API.

Estas herramientas permiten a los desarrolladores y equipos de seguridad implementar medidas efectivas para prevenir y corregir fallas en sus aplicaciones. La combinación de pruebas automatizadas, análisis de código y protección en tiempo real es esencial para mantener la seguridad de las API modernas.

2.2 Herramientas OWASP para Resolver Vulnerabilidades

OWASP ofrece diversas herramientas y recursos para mitigar estas vulnerabilidades:

- **OWASP Testing Guide:** Guía detallada para realizar pruebas de seguridad en aplicaciones web y API, que ayuda a identificar vulnerabilidades específicas.
- **OWASP Cheat Sheet Series:** Hojas informativas con recomendaciones prácticas y ejemplos de código para evitar vulnerabilidades comunes.
- **OWASP ZAP (Zed Attack Proxy):** Herramienta gratuita y de código abierto para pruebas de penetración y detección de vulnerabilidades en aplicaciones web y API.
- **OWASP Dependency-Check:** Analiza dependencias y bibliotecas para identificar componentes vulnerables y ayudar en la gestión de riesgos.
- **OWASP ModSecurity Core Rule Set:** Conjunto de reglas para firewalls de aplicaciones web que ayuda a bloquear ataques comunes y proteger las API.

Estas herramientas permiten a los desarrolladores y equipos de seguridad implementar medidas efectivas para prevenir y corregir fallas en sus aplicaciones. La combinación de pruebas automatizadas, análisis de código y protección en tiempo real es esencial para mantener la seguridad de las API modernas.

2.3 1. API1:2023 – Autorización de Nivel de Objeto Rota (BOLA)

- **Descripción:** Ocurre cuando una API no verifica adecuadamente si un usuario tiene permiso para acceder a un objeto específico, permitiendo que atacantes manipulen identificadores de objetos para acceder a datos de otros usuarios.
- **Impacto:** Acceso no autorizado a información confidencial y posibilidad de realizar acciones no autorizadas, como modificar o eliminar datos.
- **Ejemplo:** Un atacante autenticado como el usuario A accede o modifica los datos del usuario B mediante la manipulación del ID del objeto.
- **Causas Raíz y Prevención:**
  - Falta de controles de acceso a nivel de objeto.
  - Implementar políticas de acceso estrictas y controles en la capa lógica de la aplicación.
  - Realizar pruebas automatizadas para detectar fallas de autorización.

2.4 2. API2:2023 – Autenticación Rota

- **Descripción:** Implementaciones deficientes de autenticación, como contraseñas débiles o manejo inseguro de tokens.
- **Impacto:** Compromiso de cuentas de usuario, robo de datos y transacciones no autorizadas.
- **Ejemplo:** Uso de credenciales robadas o filtradas para acceder a una API.
- **Causas Raíz y Prevención:**
  - Requisitos de contraseñas débiles.
  - Implementar autenticación multifactor y políticas de contraseñas robustas.
  - Monitorear intentos de acceso y aplicar bloqueos por tasa.

2.5 3. API3:2023 – Autorización de Nivel de Propiedad de Objeto Rota

- **Descripción:** Ocurre cuando no se aplican controles adecuados en propiedades específicas de un objeto.
- **Impacto:** Escaladas de privilegios o pérdida de integridad de datos.
- **Ejemplo:** Cambiar el plan de suscripción modificando la propiedad `"account-type"`.
- **Causas Raíz y Prevención:**
  - Falta de validación de autorización a nivel de propiedad.
  - Implementar controles detallados y validaciones estrictas por propiedad.

2.6 4. API4:2023 – Consumo de Recursos sin Restricciones

- **Descripción:** Falta de limitaciones en el uso de recursos puede ser explotado para agotar el sistema.
- **Impacto:** Denegación de servicio (DoS), costos operativos altos y degradación del rendimiento.
- **Ejemplo:** Un atacante envía múltiples solicitudes que consumen muchos recursos.
- **Causas Raíz y Prevención:**
  - Ausencia de límites de tasa y cuotas.
  - Implementar mecanismos de limitación y monitoreo.
  - Usar gateways de API para control de acceso.

2.7 5. API5:2023 – Autorización de Nivel de Función Rota

- **Descripción:** Controles de acceso inadecuados en funciones específicas.
- **Impacto:** Acceso a funciones administrativas o sensibles sin autorización.

- **Ejemplo:** Un usuario accede a un endpoint de administrador sin permisos.
- **Causas Raíz y Prevención:**
  - Falta o mal implementación de controles de acceso.
  - Definir roles y permisos claramente y validarlos por función.

2.8 6. API6:2023 – Falsificación de Solicitudes del Lado del Servidor (SSRF)

- **Descripción:** La API permite que el servidor realice solicitudes a destinos maliciosos.
- **Impacto:** Acceso a recursos internos, escaneo de puertos y exfiltración de datos.
- **Ejemplo:** Solicitud maliciosa que accede a recursos internos del servidor.
- **Causas Raíz y Prevención:**
  - Falta de validación de URLs proporcionadas.
  - Usar listas blancas y validar estrictamente las URLs.

2.9 7. API7:2023 – Configuración de Seguridad Incorrecta

- **Descripción:** Configuraciones inseguras o por defecto en la API o servidores.
- **Impacto:** Exposición de datos sensibles, acceso no autorizado y explotación de vulnerabilidades.
- **Ejemplo:** API que expone información de depuración o usa credenciales por defecto.
- **Causas Raíz y Prevención:**
  - Uso de configuraciones inseguras o por defecto.
  - Realizar auditorías, aplicar principios de configuración segura.

2.10 8. API8:2023 – Falta de Protección Frente a Amenazas Automatizadas

- **Descripción:** Ausencia de protección contra bots o ataques automatizados.
- **Impacto:** Explotación de flujos de negocio, fraude y agotamiento de recursos.
- **Ejemplo:** Bots compran entradas para revenderlas, afectando usuarios reales.
- **Causas Raíz y Prevención:**
  - No detección de tráfico automatizado.
  - Implementar detección de bots, CAPTCHA y límites de velocidad.

2.11 9. API9:2023 – Gestión Inadecuada del Inventario

- **Descripción:** Falta de control y documentación sobre versiones y endpoints de las APIs.
- **Impacto:** Exposición de datos por uso de versiones obsoletas o endpoints no documentados.
- **Ejemplo:** API antigua expone datos por falta de mantenimiento.
- **Causas Raíz y Prevención:**
  - Ausencia de políticas de versionado y retiro.
  - Gestionar activos y auditar periódicamente.

2.12 10. API10:2023 – Consumo No Seguro de APIs

- **Descripción:** API que consume servicios de terceros sin validar respuestas.
- **Impacto:** Violación de privacidad, robo de datos y control de cuentas.
- **Ejemplo:** Redirección a sitios maliciosos por una API comprometida.
- **Causas Raíz y Prevención:**
  - Validación insuficiente de respuestas de terceros.
  - Evaluar seguridad de APIs externas, validar y cifrar toda la comunicación.

3 Metodología

1. **Investigación Inicial:** Revisión de la documentación oficial de OWASP API Security Top 10 - 2023 y análisis de las vulnerabilidades identificadas.
2. **Análisis de Casos de Uso:** Identificación de escenarios comunes donde las vulnerabilidades de API pueden ser explotadas.
3. **Implementación de Pruebas:** Desarrollo de pruebas específicas para cada vulnerabilidad identificada.
4. **Validación de Herramientas:** Prueba y evaluación de las herramientas OWASP mencionadas para la detección de vulnerabilidades.
5. **Documentación de Resultados:** Registro detallado de los hallazgos y recomendaciones de seguridad.

4 Desarrollo

4.1 Implementación

4.1.1 Paso 1: Configuración del Entorno de Prueba

Figura 1: Arquitectura del entorno de prueba

4.1.2 Paso 2: Implementación de Pruebas de Seguridad

- Configuración de OWASP ZAP para pruebas de API
- Implementación de pruebas automatizadas para cada vulnerabilidad
- Configuración de OWASP Dependency-Check para análisis de dependencias
- Implementación de reglas de ModSecurity para protección de API

4.2 Evaluación de Resultados

- Análisis detallado de los resultados obtenidos
- Identificación de vulnerabilidades encontradas

- Evaluación de la efectividad de las herramientas OWASP
- Documentación de los casos de éxito y fracaso

4.3 Análisis y Discusión Sobre los Resultados

- Discusión de las vulnerabilidades más críticas encontradas
- Análisis de las fortalezas y debilidades de las herramientas OWASP
- Impacto de las vulnerabilidades en diferentes escenarios de uso
- Recomendaciones específicas para mitigar cada vulnerabilidad

5 Conclusiones

- Las vulnerabilidades de API son una amenaza real y creciente
- Las herramientas OWASP son efectivas para detectar y mitigar vulnerabilidades
- La implementación de múltiples capas de seguridad es esencial
- La educación y concienciación sobre seguridad de API es necesaria
- La combinación de herramientas automatizadas y análisis manual es la mejor aproximación

6 Recomendaciones

- Implementar controles de acceso estrictos en todas las API
- Utilizar herramientas de análisis estático y dinámico de código
- Mantener un inventario actualizado de APIs y sus versiones
- Implementar monitoreo y logging continuo de las API
- Realizar pruebas de seguridad periódicas
- Mantener actualizadas todas las dependencias y bibliotecas

7 Referencias

- OWASP. (2023). OWASP API Security Top 10 - 2023. https://owasp.org/www-project-api-security/
- OWASP. (2023). OWASP Zed Attack Proxy Project. https://www.zaproxy.org/
- OWASP. (2023). OWASP Dependency-Check. https://owasp.org/www-project-dependency-check/
- OWASP. (2023). OWASP ModSecurity Core Rule Set Project. https://owasp.org/www-project-modsecurity-core-rule-set/
- OWASP. (2023). OWASP Testing Guide v5. https://owasp.org/www-project-web-security-testing-guide/