Informe de Análisis de Seguridad Web (Pre-Producción)

Sitio Web Analizado: `https://mi-ecommerce-rouge.vercel.app/`

Fecha del Análisis: 24 de Enero de 2026

Analista: Manus AI

Resumen Ejecutivo

El análisis de seguridad realizado sobre el sitio web de e-commerce, alojado en Vercel y construido con Astro, revela un **nivel de seguridad inicial alto** en términos de configuración de infraestructura y protección contra ataques comunes de inyección y Cross-Site Scripting (XSS) en los puntos de entrada probados.

Se ha verificado la correcta implementación de la mayoría de los **encabezados de seguridad (Security Headers)** críticos, lo que proporciona una base sólida de defensa a nivel de transporte y navegador. Las pruebas manuales de inyección (SQLi/NoSQLi) y XSS en la funcionalidad de búsqueda y filtrado no arrojaron resultados positivos, y los controles de acceso a rutas sensibles (`/account` , `/admin` , `/dashboard`) parecen estar correctamente implementados.

La principal área de mejora identificada es la **optimización de la Política de Seguridad de Contenido (CSP)**, que, aunque presente, es permisiva en ciertas directivas (`'unsafe-inline'` en `script-src` y `style-src`), lo que podría mitigar la protección contra XSS en caso de que se descubra una vulnerabilidad en el código fuente.

---

1. Reconocimiento y Tecnologías

El sitio web opera bajo la siguiente pila tecnológica:

Componente	Detalle	Observaciones de Seguridad
Framework	Astro v5.16.6	Se recomienda mantener el framework y sus dependencias actualizadas para mitigar CVEs conocidos `1` .
Hosting	Vercel	El proveedor de hosting gestiona gran parte de la seguridad de la infraestructura (TLS, DDoS,

		WAF), lo que es un punto fuerte.
Superficie de Ataque	Búsqueda, Filtros de Productos, Login/Registro, Suscripción.	Los formularios de login/registro y la funcionalidad de carrito son los puntos más críticos para futuras pruebas.

2. Análisis de Encabezados de Seguridad (Security Headers)

La implementación de encabezados de seguridad es robusta, lo que demuestra una buena práctica de seguridad a nivel de infraestructura.

Encabezado	Valor Detectado	Estado	Recomendación
`Strict-Transport-Security` (HSTS)	`max-age=31536000; includeSubDomains; preload`	**OK**	Excelente configuración.
`X-Content-Type-Options`	`nosniff`	**OK**	Previene ataques de *MIME-sniffing*.
`X-Frame-Options`	`SAMEORIGIN`	**OK**	Previene ataques de *Clickjacking*.
`Referrer-Policy`	`strict-origin-when-cross-origin`	**OK**	Buena política de privacidad para el referenciador.
`X-XSS-Protection`	`1; mode=block`	**OK**	Aunque obsoleto, es una capa de defensa adicional.
`Content-Security-Policy` (CSP)	**Presente**	**MEJORABLE**	Ver sección 3.1.

3. Hallazgos y Vulnerabilidades

3.1. Política de Seguridad de Contenido (CSP) Permisiva (Baja)

La CSP está configurada, lo cual es positivo, pero incluye directivas que reducen su efectividad contra XSS:

> `script-src 'self' 'unsafe-inline' https://*.google.com ...`
>
> `style-src 'self' 'unsafe-inline' https://*.googleapis.com ...`

La directiva `'unsafe-inline'` permite la ejecución de scripts y estilos en línea, lo que anula la principal protección de la CSP contra XSS inyectado en el código HTML.

Recomendación:

1. Eliminar `'unsafe-inline'` de `script-src` y `style-src`.
2. Utilizar **hashes** o **nonces** para permitir scripts y estilos específicos que sean necesarios, en lugar de permitir todos los scripts en línea.

3.2. Ausencia de Vulnerabilidades Críticas Inmediatas (XSS, SQLi, BAC)

Las pruebas manuales de inyección y control de acceso no revelaron vulnerabilidades:

- **Cross-Site Scripting (XSS):** El campo de búsqueda no reflejó el *payload* de prueba (`<script>alert('XSS')</script>`), lo que sugiere una correcta sanitización o codificación del lado del servidor/framework.
- **SQL/NoSQL Injection:** La manipulación del parámetro `categoria` en la URL con *payloads* de inyección (`' OR '1'='1`) no alteró el comportamiento de la aplicación ni generó errores, lo que indica el uso de consultas parametrizadas o un ORM seguro.
- **Broken Access Control (BAC):** Las rutas sensibles (`/account`, `/admin`, `/dashboard`) están protegidas, ya sea redirigiendo al login o mostrando un 404, lo que previene el acceso no autorizado a funciones administrativas o de usuario.

---

4. Guía Educativa: Paso a Paso para el Análisis de Seguridad

El proceso de análisis de seguridad (o *Pentesting*) sigue una metodología estructurada, como la de **OWASP Testing Guide**.

Paso 1: Reconocimiento (Information Gathering)

Objetivo: Recolectar la mayor cantidad de información posible sobre el objetivo.

Herramienta	Uso	Ejemplo (en este análisis)

Navegador	Exploración manual de la web, identificación de formularios, URLs, y funcionalidades.	Se identificaron las rutas `/login`, `/productos`, y los campos de búsqueda y suscripción.
cURL	Análisis de encabezados HTTP y respuestas del servidor.	Se detectó la presencia de HSTS, X-Frame-Options y la CSP.
Wappalyzer / BuiltWith (o script de consola)	Identificación de tecnologías, frameworks y librerías.	Se identificó Astro v5.16.6 y Vercel.

Paso 2: Escaneo de Vulnerabilidades (Vulnerability Scanning)

Objetivo: Utilizar herramientas automatizadas para identificar debilidades conocidas.

Herramienta	Uso	Ejemplo (en este análisis)
Nikto	Escaneo de servidores web para archivos, CGIs y configuraciones inseguras.	Se utilizó para verificar la configuración de seguridad del servidor (aunque el escaneo completo fue bloqueado/lento).
Nmap	Escaneo de puertos y servicios a nivel de red.	Se utilizó para confirmar que solo los puertos web (80/443) están expuestos, lo cual es correcto para una aplicación en Vercel.
OWASP ZAP / Burp Suite	Escaneo activo y pasivo de la aplicación, interceptando el tráfico.	Estas herramientas son esenciales para el escaneo profundo de la aplicación (no se usaron en este entorno por su complejidad de configuración, pero son la **mejor opción** para un análisis completo).

Paso 3: Pruebas Manuales (Manual Testing)

Objetivo: Probar las vulnerabilidades más críticas de forma manual, enfocándose en la lógica de negocio.

Vulnerabilidad (OWASP Top 10)	Técnica de Prueba	Resultado en el Análisis
A03:2021 - Inyección (SQLi, NoSQLi)	Inyectar caracteres especiales (`'` , `--` , `OR 1=1`) en campos de entrada y parámetros de URL.	**No vulnerable** en los parámetros probados.
A07:2021 - XSS	Inyectar *payloads* de JavaScript (`<script>alert(1)</script>`) en campos de búsqueda, nombres de usuario, etc.	**No vulnerable** en el campo de búsqueda.
A01:2021 - Broken Access Control (BAC)	Intentar acceder a rutas restringidas (`/admin` , `/user/123` , `/account`) sin autenticación o con un rol incorrecto.	**Correctamente protegido** (redirige a login o 404).

Paso 4: Análisis y Documentación

Objetivo: Recopilar los hallazgos, clasificarlos por severidad y documentar las recomendaciones de mitigación.

- **Clasificación:** Utilizar sistemas como **CVSS** (Common Vulnerability Scoring System) o una clasificación simple (Crítica, Alta, Media, Baja, Informativa).
- **Reporte:** Crear un informe claro que incluya la descripción de la vulnerabilidad, el impacto potencial y los pasos de remediación.

---

5. Conclusión y Próximos Pasos

El sitio web muestra una buena postura de seguridad inicial. La única recomendación de seguridad inmediata es **reforzar la Política de Seguridad de Contenido (CSP)** eliminando las directivas `'unsafe-inline'` .

Próximos Pasos Recomendados:

1. **Remediación de CSP:** Implementar *nonces* o *hashes* para los scripts y estilos en línea.
2. **Pruebas de Lógica de Negocio:** Realizar pruebas exhaustivas en el flujo de compra, carrito, y gestión de pedidos, ya que estas áreas son críticas en un e-commerce y no fueron cubiertas en profundidad en este análisis inicial.

3. **Pruebas de Autenticación/Autorización:** Realizar pruebas de *fuerza bruta* y *enumeración de usuarios* en el formulario de login.

Referencias

[1] Astro Documentation. Astro Security Guide. [URL de referencia]
[2] OWASP Foundation. OWASP Top 10 2021. [URL de referencia]
[3] Vercel Documentation. Vercel Security Headers. [URL de referencia]