Sanitización de Metadatos YAML: Prevención XSS y Rotura del DOM

Estantería: Desarrollo y Arquitectura

Subtema: Seguridad

mercedev.es — 2026-05-17 | Epic 3 - Fase 1

El Desafío (Síntoma)

Durante la generación del índice de la Biblioteca, la página estática dejó de cargar repentinamente y el contenido inferior desapareció. El problema se originó porque el campo `descripcion` de un artículo contenía etiquetas HTML literales (como `<script src="...">`). Al inyectarse este texto "crudo" en la plantilla del SSG, el navegador lo interpretó como código ejecutable real, rompiendo la estructura del DOM al carecer de etiquetas de cierre adecuadas. Adicionalmente, esto abría una brecha de seguridad para ataques de Cross-Site Scripting (XSS).

La Maniobra (Lógica)

Se implementó un patrón de sanitización estricta ("Shift-Left Security") en los orquestadores `merci-publish.py` y `merci-wp.py`. Se utilizó la función nativa `html.escape()` de Python para procesar todos los campos de texto menores extraídos del YAML Frontmatter (título, descripción, fase, tipo, volumen, fecha) antes de su interpolación en las plantillas f-string de HTML y en la compilación de PDFs.

El Aprendizaje / Deuda Técnica

Confiar ciegamente en las entradas de datos, incluso cuando provienen de archivos Markdown locales redactados por el propio equipo, es un antipatrón arquitectónico. Las etiquetas literales pueden destruir interfaces visuales o romper generadores de PDF dependientes de etiquetas (como WeasyPrint). Sanitizar en tiempo de compilación (Build Time) cierra permanentemente la vía de inyección secundaria, demostrando que la seguridad debe aplicarse desde la extracción del dato y no solo en el entorno de ejecución del cliente.