Prevención de Fuga de Datos (DLP) en la creación de Boilerplates

Cuadernillo | Vol. 1

mercedev.es — 2026-04-30 | Fase 8 (Expansión de Contenido)

El Desafío (Síntoma)

Durante el empaquetado del repositorio matriz (`mercedev.es`) hacia su versión pública redistribuible (`merci-boilerplate`), surgió la necesidad de preservar el andamiaje del CMS Headless (carpetas vacías como `blog/` y `art-de-cote/`). La primera iteración del script de instanciación simplemente excluía estas carpetas del proceso de borrado. Esto generó un vector crítico de Fuga de Datos (Data Leak): cualquier borrador privado o artículo publicado almacenado en dichas carpetas en la máquina del desarrollador viajaría intacto al repositorio público de GitHub.

La Maniobra (Lógica)

En lugar de usar listas de exclusión (Blacklisting), se rediseñó el destructor (`merci-init.py`) aplicando el patrón de **Destrucción y Recreación**: 1. **Purga Total:** Se arrasa con los directorios dinámicos completos usando `shutil.rmtree()` , garantizando la erradicación de cualquier propiedad intelectual. 2. **Reconstrucción Inmaculada:** Inmediatamente después, el script vuelve a crear las carpetas desde cero (`mkdir`) e inyecta dinámicamente archivos ocultos `.gitkeep` .

El Aprendizaje / Deuda Técnica

Un Boilerplate público debe operar bajo el principio de "Zero Trust" (Confianza Cero) respecto al contenido del repositorio matriz.

Intentar salvar carpetas vacías omitiéndolas de una purga es un antipatrón arquitectónico. Git no rastrea directorios vacíos; rastrea archivos. Al obligar al script a destruir y luego regenerar programáticamente la topología estructural junto con sus marcadores `.gitkeep` , garantizamos que el nuevo usuario reciba un

ecosistema Headless funcional, mientras nuestra propiedad intelectual queda matemáticamente bloqueada antes del primer commit público.