Despliegue SSG Puro con GitHub Actions (rsync)

Cuadernillo | Vol. 1

mercedev.es — 2026-05-05

El Desafío (Síntoma)

Durante la Fase 11 de maduración del ecosistema, se buscó automatizar el despliegue del código hacia el servidor de producción utilizando CI/CD (Continuous Integration / Continuous Deployment - Integración Continua / Despliegue Continuo). El objetivo era compilar el SSG (Static Site Generation - Generación de Sitios Estáticos) en la nube y enviar los archivos listos a través de SSH (Secure Shell).

Sin embargo, al aplicar el comando `rsync --delete`, se detectó que la capa dinámica del CMS (Content Management System - Sistema de Gestión de Contenidos) en producción dejaba de funcionar. `rsync` purgaba implacablemente el enlace simbólico físico (`public/blog`) que unía el núcleo estático con la instalación asilada de WordPress. Como dicho enlace está excluido en el `.gitignore`, el robot de GitHub asumía que era un archivo "sobrante" en el servidor y lo destruía.

La Maniobra (Lógica)

En lugar de sobreingeniar el comando `rsync` con exclusiones complejas para mantener vivo el puente del CMS, se tomó la decisión arquitectónica de **hacer un rollback y descartar el despliegue automático para esta infraestructura**. La arquitectura híbrida mantiene el control manual mediante `git pull` en el servidor, el cual es lo suficientemente inteligente para ignorar los archivos no rastreados y preservar la integración.

No obstante, el código del flujo desarrollado es una pieza de ingeniería robusta y validada. Superó las barreras de permisos e inyectó llaves de forma inmaculada. Se archiva a continuación como un activo "Plug & Play" para su uso directo en futuras arquitecturas que sean **100% estáticas** (sin CMS acoplado).

```yaml
 name: Build & Deploy (SSG Cloud)

on:
  push:
    branches: [ main ]
    paths:
      - '.github/workflows/**'
      - 'biblioteca/**'
      - 'public/**'
      - 'scripts/merci/**'
      - 'src/scss/**'

env:
  FORCE_JAVASCRIPT_ACTIONS_TO_NODE24: true

jobs:
  deploy:
    runs-on: ubuntu-latest
    environment: 'Droplet Producción mercedev-php'
    steps:
      - name: Checkout repository
        uses: actions/checkout@v4

      - name: Setup Python
        uses: actions/setup-python@v5
        with:
          python-version: '3.10'

      - name: Execute SSG Compilation (Build)
        run: |
          python3 scripts/merci/merci-publish.py
          python3 scripts/merci/merci-sync-pages.py

      - name: Deploy to Production (CloudPanel) via rsync
        env:
          SSH_PRIVATE_KEY: ${{ secrets.DEPLOY_SSH_KEY }}
          SERVER_IP: ${{ secrets.SERVER_IP }}
          SERVER_USER: ${{ secrets.SERVER_USER }}
        run: |
```

```
          if [ -z "$SERVER_IP" ] || [ -z "$SERVER_USER" ]; then
            echo "🚨 ERROR: Faltan secretos. Revisa SERVER_IP o
SERVER_USER en GitHub Settings > Secrets."
            exit 1
          fi

          # Inicializar el agente SSH nativo en memoria RAM
          eval $(ssh-agent -s)

          mkdir -p ~/.ssh
          echo "$SSH_PRIVATE_KEY" > ~/.ssh/id_ed25519
          chmod 600 ~/.ssh/id_ed25519

          # Cargar la llave auditada en la memoria del agente
          ssh-add ~/.ssh/id_ed25519

          # Añadir la firma del servidor y desplegar confiando en el
agente
          ssh-keyscan -H $SERVER_IP >> ~/.ssh/known_hosts 2>/dev/null
          rsync -avz -e "ssh -v -o StrictHostKeyChecking=no" --delete
public/ $SERVER_USER@$SERVER_IP:~/htdocs/mercedev.es/public/
```

El Aprendizaje / Deuda Técnica

El código que no llega a producción pero ha sido empíricamente validado es I+D puro. Tirarlo a la basura es desperdiciar esfuerzo de ingeniería de sistemas.

Lección Arquitectónica: Las herramientas destructivas de sincronización (`rsync --delete`) son por defecto hostiles contra arquitecturas híbridas que mantienen estado local u oculto en el servidor de destino (como enlaces simbólicos o carpetas de subidas de usuarios que no viajan en Git). Para ecosistemas acoplados, la sincronización pasiva (`git pull`) o la containerización inmutable (Docker) resultan patrones sustancialmente más seguros.