EcomLabs

GUÍA PASO A PASO

El día del parche de Next.js

Qué dejar listo antes del parche de seguridad del 3 de noviembre, cómo actualizar con npm, pnpm, yarn o bun sin romper el lockfile, qué probar antes de desplegar y cómo revertir si algo falla.

PARA	Proyectos en producción con Next.js 16 o 15
PROBADO CON	Next.js 15.5.27 → 15.5.28 · Node 22.22.3 · npm 11.11.1 · pnpm 10.2.0 · Windows 11
ACTUALIZADA	2 de noviembre de 2026
DOCUMENTACIÓN	nextjs.org/blog/security
ARTÍCULO	ecomlabs.dev/g/parche
AUTOR	Luis Morales · @luismoralesdev

Índice

Antes de empezar	Qué cubre esta guía
Paso 01	Hoy: anota la versión que corre
Paso 02	Hoy: lockfile, CI, pruebas y rama
Paso 03	El día del parche: lee y actualiza
Paso 04	Prueba y despliega
Paso 05	Si algo se rompe: rollback
Al terminar	Checklist

ANTES DE EMPEZAR

Qué cubre esta guía

Next.js publica el 3 de noviembre de 2026 un parche de seguridad para siete vulnerabilidades de severidad media. Cuando escribimos esta guía, el 2 de noviembre, no había versiones ni CVE publicados, así que aquí no hay números inventados: donde veas `<v>` va la versión que indique el post del parche para tu línea (16.x o 15.x).

Probamos los comandos de npm y pnpm el 2 de noviembre, actualizando de Next.js 15.5.27 a 15.5.28 (la versión con parche del 14 de octubre) con Node 22.22.3, npm 11.11.1 y pnpm 10.2.0 en Windows 11. Los de Yarn y Bun vienen de su documentación: no los ejecutamos.

PASO 01

Hoy: anota la versión que corre

En la raíz del proyecto, con npm:

```
npm ls next
npx next --version
npm view next dist-tags.latest
npm view next dist-tags.backport
```

Con los otros gestores: `pnpm list next` (o `pnpm why next`), `yarn why next` o `bun why next`. `npx next --version` sirve con cualquiera.

- Anota la versión del build que está en producción y su commit, no la de tu rama. Corre `next --version` también donde se hace el build.
- Hoy `latest` es 16.4.4, una versión que solo corrige bugs: no es el parche. En 15.x tu referencia es `backport` (hoy, 15.5.28).
- `next info` compara con `latest` y marca como desactualizada cualquier 15.x, aunque tenga parche. No lo uses para decidir.
- En un monorepo, `npm ls next` puede mostrar varias copias. Cuentan todas.

PASO 02

Hoy: lockfile, CI, pruebas y rama

- [ ] El lockfile está en git: `package-lock.json`, `pnpm-lock.yaml`, `yarn.lock` o `bun.lock`.
- [ ] El CI instala sin reescribirlo: `npm ci`, `pnpm install --frozen-lockfile`, `yarn install --immutable` (Yarn 1: `--frozen-lockfile`), `bun install --frozen-lockfile` o `bun ci`.
- [ ] El build y las pruebas pasan hoy en la rama principal.
- [ ] Si usas Image Optimization (`/_next/image`), Server Actions, middleware o proxy, `rewrites()` o la caché (ISR, `use cache`), tienes una prueba o un paso del smoke test que pasa por ahí: son las zonas que tocaron los avisos de 2026.

- [ ] El smoke test recorre tus flujos críticos: inicio, producto, carrito, checkout, inicio de sesión.
- [ ] Decidiste el nombre de la rama (por ejemplo, `chore/next-security-2026-11`) y quién aprueba el merge.
- [ ] Sabes volver a desplegar el build anterior en tu plataforma y ya lo ensayaste.

PASO 03

El día del parche: lee y actualiza

1. Abre `nextjs.org/blog/security` y el post del parche. Lee cada aviso (GHSA y CVE): versiones afectadas y a qué configuraciones afecta.
2. Elige la versión de tu línea: la 16 si estás en 16.x, la 15 si estás en 15.x. No cambies de versión mayor el día del parche.
3. Confirma que es la versión del post y no una versión de bugs publicada el día anterior, como 16.4.4 el 2 de noviembre o 16.3.7 el 29 de septiembre.

GESTOR	ACTUALIZAR	INSTALAR EN CI
npm	`npm install next@<v>`	`npm ci`
pnpm	`pnpm add next@<v>`	`pnpm install --frozen-lockfile`
Yarn 2+	`yarn add next@<v>`	`yarn install --immutable`
Yarn 1	`yarn add next@<v>`	`yarn install --frozen-lockfile`
Bun	`bun add next@<v>`	`bun install --frozen-lockfile`

El flujo con npm, en una rama:

```
git switch -c chore/next-security-2026-11
npm install next@<v>
git diff --stat
npm run build
git commit -am "chore(deps): next <v>"
```

- El diff debe tocar solo `package.json` y el lockfile. En el lockfile cambian `next`, `@next/env` y los binarios `@next/swc-*`: en nuestra prueba, 10 entradas. Si toca cientos de paquetes, alguien regeneró el lockfile.
- Si usas `eslint-config-next`, súbelo al mismo número, como hace `create-next-app`: `npm install next@<v> eslint-config-next@<v>`.
- npm guarda `"^<v>"` en `package.json`; pnpm 10, en nuestra prueba, guardó la versión exacta. Lo que fija la versión desplegada es el lockfile.
- Si el aviso nombra otro paquete, compruébalo con `npm ls <paquete>`. En octubre el aviso pedía `sharp` 0.35.5 o posterior, pero next@15.5.28 acepta `^0.34.3 || ^0.35.5`: en nuestra prueba, con un lockfile que traía 0.34.5, npm y pnpm lo conservaron. `npm update sharp` (o `pnpm update sharp`) lo subió a 0.35.5.

PASO 04

Prueba y despliega

- [ ] `npm run build` pasa y su primera línea muestra la versión nueva.
- [ ] Las pruebas pasan en CI con la instalación congelada.
- [ ] El smoke test pasa en un preview o en staging, con las variables de entorno de producción.
- [ ] Las zonas de los avisos que usas responden igual que antes: imágenes, formularios con Server Actions, rutas protegidas por middleware o proxy, páginas en caché.
- [ ] Los logs del preview no muestran errores nuevos.

El mínimo, en local, después del build:

```
npx next start -p 3123
curl -s -o /dev/null -w "%{http_code}\n" http://localhost:3123/
```

Corre `curl` en otra terminal; en nuestra prueba respondió `200`. Después del deploy, confirma la versión que corre en producción y vuelve a leer el post esa tarde: en septiembre, el aviso recibió notas nuevas el mismo día del parche.

PASO 05

Si algo se rompe: rollback

1. Vuelve a desplegar el build anterior desde tu plataforma. Es lo más rápido y no depende de un build nuevo.
2. Deja el repo igual que producción con un revert del commit del parche, y reinstala desde el lockfile:

```
git revert --no-edit HEAD
npm ci
npm ls next
```

En nuestra prueba, `npm ls next` volvió a mostrar 15.5.27. Si el CI falla con `npm ci` porque `package.json` y el lockfile no coinciden, el arreglo es regenerar el lockfile con `npm install` y hacer commit de los dos, no cambiar `npm ci` por `npm install`.

> **CUIDADO**
> Revertir un parche de seguridad te deja expuesto otra vez. Úsalo para ganar tiempo: arregla lo que se rompió y vuelve a desplegar la versión con parche. Una regla de firewall del hosting es una mitigación temporal, no un sustituto del parche.

AL TERMINAR

Checklist

- [ ] Anotaste la versión que corre en producción y el commit desplegado.

- [ ] El lockfile está en git y el CI instala congelado.
- [ ] Leíste el post del parche y cada aviso antes de tocar nada.
- [ ] Actualizaste a la versión de tu línea, en una rama, sin cambiar de versión mayor.
- [ ] El diff solo mueve `next`, `@next/env`, `@next/swc-*`, `eslint-config-next` si lo usas y los paquetes que nombre el aviso.
- [ ] Comprobaste con `npm ls` cada paquete que nombre el aviso, no solo `next`.
- [ ] Build, pruebas y smoke test pasaron en preview.
- [ ] Confirmaste la versión en producción después del deploy.
- [ ] Sabes cómo volver al build anterior si algo falla.