EcomLabs

STEP-BY-STEP GUIDE

Next.js security release day

What to have ready before the November 3 security release, how to upgrade with npm, pnpm, yarn or bun without breaking the lockfile, what to test before you deploy, and how to roll back if something fails.

FOR	Production projects on Next.js 16 or 15
TESTED WITH	Next.js 15.5.27 → 15.5.28 · Node 22.22.3 · npm 11.11.1 · pnpm 10.2.0 · Windows 11
UPDATED	November 2, 2026
DOCS	nextjs.org/blog/security
ARTICLE	ecomlabs.dev/g/parche-en
AUTHOR	Luis Morales · @luismoralesdev

Contents

Before you start	What this guide covers
Step 01	Today: write down the version that is running
Step 02	Today: lockfile, CI, tests and branch
Step 03	Release day: read, then upgrade
Step 04	Test, then deploy
Step 05	If something breaks: roll back
When you finish	Checklist

BEFORE YOU START

What this guide covers

Next.js ships a security release on November 3, 2026 for seven medium-severity vulnerabilities. When we wrote this guide, on November 2, no versions or CVEs had been published, so there are no made-up numbers here: wherever you see `<v>`, use the version the release post gives for your line (16.x or 15.x).

We tested the npm and pnpm commands on November 2, upgrading from Next.js 15.5.27 to 15.5.28 (the patched version from October 14) with Node 22.22.3, npm 11.11.1 and pnpm 10.2.0 on Windows 11. The Yarn and Bun commands come from their docs: we did not run them.

STEP 01

Today: write down the version that is running

From the project root, with npm:

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

With the other package managers: `pnpm list next` (or `pnpm why next`), `yarn why next` or `bun why next`. `npx next --version` works with any of them.

- Write down the version of the build in production and its commit, not the one on your branch. Run `next --version` where the build runs too.
- Today `latest` is 16.4.4, a bug-fix-only release: it is not the security release. On 15.x your reference is the `backport` tag (15.5.28 today).
- `next info` compares against `latest` and flags every 15.x as outdated, patched or not. Do not use it to decide.
- In a monorepo, `npm ls next` can show several copies. All of them count.

STEP 02

Today: lockfile, CI, tests and branch

- [ ] The lockfile is committed: `package-lock.json`, `pnpm-lock.yaml`, `yarn.lock` or `bun.lock`.
- [ ] CI installs without rewriting it: `npm ci`, `pnpm install --frozen-lockfile`, `yarn install --immutable` (Yarn 1: `--frozen-lockfile`), `bun install --frozen-lockfile` or `bun ci`.
- [ ] The build and the tests pass today on main.
- [ ] If you use Image Optimization (`/_next/image`), Server Actions, middleware or proxy, `rewrites()` or caching (ISR, `use cache`), a test or a smoke-test step goes through it: those are the areas the 2026 advisories touched.
- [ ] The smoke test walks your critical flows: home, product, cart, checkout, sign-in.

- [ ] You picked the branch name (for example `chore/next-security-2026-11`) and who approves the merge.
- [ ] You know how to redeploy the previous build on your platform and have rehearsed it.

STEP 03

Release day: read, then upgrade

1. Open `nextjs.org/blog/security` and the release post. Read every advisory (GHSA and CVE): affected versions and which setups are affected.
2. Pick the version for your line: the 16 one on 16.x, the 15 one on 15.x. Do not change major versions on release day.
3. Make sure it is the version in the post and not a bug-fix release from the day before, like 16.4.4 on November 2 or 16.3.7 on September 29.

PACKAGE MANAGER	UPGRADE	INSTALL IN 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`

The npm flow, on a branch:

```
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>"
```

- The diff should only touch `package.json` and the lockfile. In the lockfile, `next`, `@next/env` and the `@next/swc-*` binaries move: 10 entries in our test. If it touches hundreds of packages, someone regenerated the lockfile.
- If you use `eslint-config-next`, move it to the same number, as `create-next-app` does: `npm install next@<v> eslint-config-next@<v>`.
- npm saves `"^<v>"` in `package.json`; pnpm 10 saved the exact version in our test. The lockfile is what pins the deployed version.
- If the advisory names another package, check it with `npm ls <package>`. In October the advisory asked for `sharp` 0.35.5 or later, but next@15.5.28 accepts `^0.34.3 || ^0.35.5`: in our test, with a lockfile that had 0.34.5, npm and pnpm kept it. `npm update sharp` (or `pnpm update sharp`) moved it to 0.35.5.

STEP 04

Test, then deploy

- [ ] `npm run build` passes and its first line shows the new version.
- [ ] The tests pass in CI with the frozen install.
- [ ] The smoke test passes on a preview or staging deployment, with production environment variables.
- [ ] The advisory areas you use behave as before: images, Server Action forms, routes protected by middleware or proxy, cached pages.
- [ ] The preview logs show no new errors.

The local minimum, after the build:

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

Run `curl` in a second terminal; in our test it answered `200`. After the deploy, confirm the version running in production and re-read the post later that day: in September the notice got new notes on release day itself.

STEP 05

If something breaks: roll back

1. Redeploy the previous build from your platform. It is the fastest path and does not depend on a new build.
2. Bring the repo back in line with production by reverting the patch commit, then reinstall from the lockfile:

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

In our test, `npm ls next` showed 15.5.27 again. If CI fails on `npm ci` because `package.json` and the lockfile disagree, the fix is to regenerate the lockfile with `npm install` and commit both, not to swap `npm ci` for `npm install`.

> **CAUTION**
> Reverting a security release leaves you exposed again. Use it to buy time: fix what broke and redeploy the patched version. A firewall rule from your host is a stopgap mitigation, not a replacement for the upgrade.

WHEN YOU FINISH

Checklist

- [ ] You wrote down the version running in production and the deployed commit.
- [ ] The lockfile is committed and CI installs frozen.

- [ ] You read the release post and every advisory before touching anything.
- [ ] You upgraded to your line's version, on a branch, without changing major versions.
- [ ] The diff only moves `next`, `@next/env`, `@next/swc-*`, plus `eslint-config-next` if you use it and any package the advisory names.
- [ ] You checked with `npm ls` every package the advisory names, not just `next`.
- [ ] Build, tests and smoke test passed on a preview.
- [ ] You confirmed the version in production after the deploy.
- [ ] You know how to get back to the previous build if something fails.