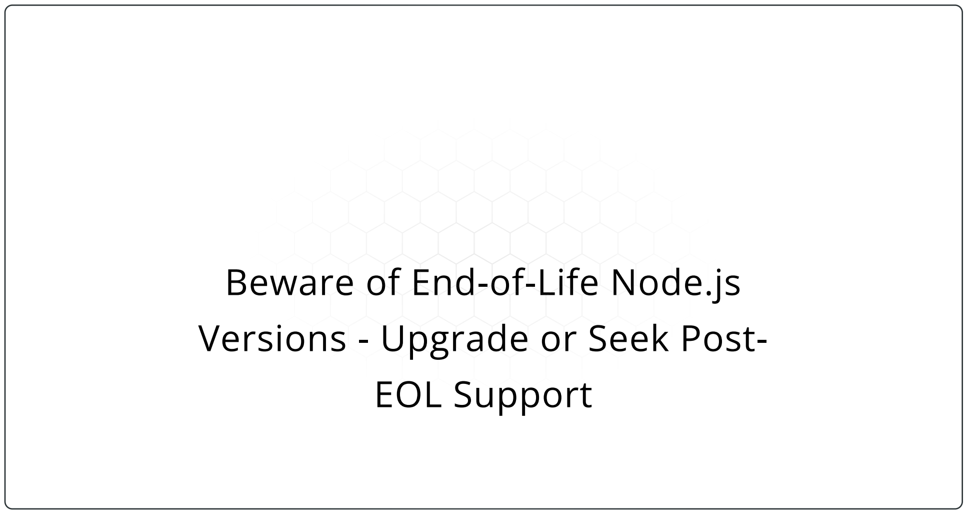
Beware of End-of-Life Node.js Versions - Upgrade or Seek Post-EOL Support

Matteo Collina


Beware of End-of-Life Node.js
Versions - Upgrade or Seek Post-
EOL Support

Beware of End-of-Life Node.js Versions *Upgrade or Seek Post-EOL Support*

The Node.js ecosystem is at a critical juncture. With Node.js 18 becoming unsupported, millions of developers need to make the jump, but they should skip 20 entirely and go straight to Node.js 22. The numbers tell a compelling story about why this upgrade isn't just recommended, it's essential. If you can't upgrade, check out our Ecosystem Sustainability Program partner, HeroDevs, which offers post End-Of-Life Node.js support.

The Support Landscape Has Changed—And Security Issues Are Real

Node.js 18 and all earlier versions are End-Of-Life. They are

Last Updated

Jun 06, 2025

Reading Time

5 min

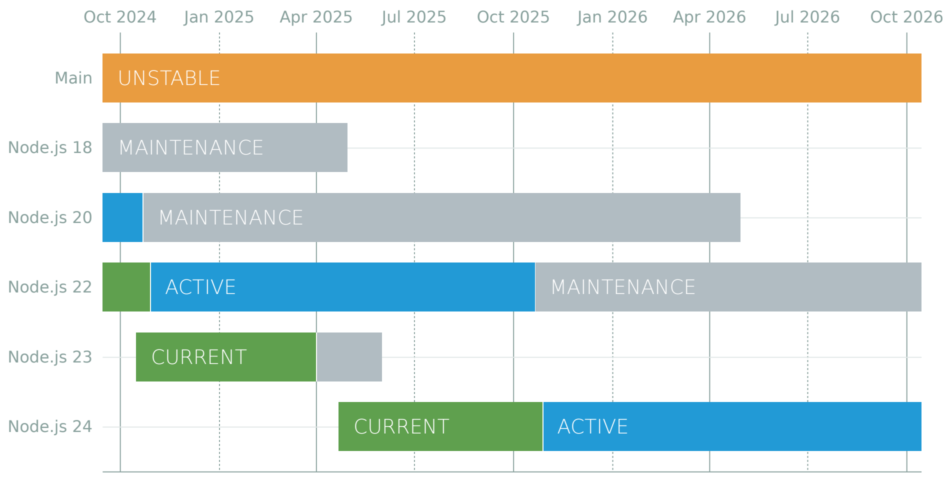
Contribute

Edit this page

Table of Contents

now completely unsupported, meaning they receive no updates, including security patches.

The security implications are immediate and serious. The May 2025 security releases revealed that Node.js 20 is vulnerable to 1 low severity issue, 1 high severity issue, and 1 medium severity issue. As the security advisory notes, "End-of-Life versions are always affected when a security release occurs", meaning Node.js 18 and all earlier versions have these same vulnerabilities but will never receive patches. Here is our release schedule:



Many ask, “Why does the Node.js project not fix vulnerabilities for all releases?”. Because it would be an ever-growing task, and some vulnerabilities could not even be fixed because they depend on a multitude of other patches to be applied. The work is simply too much, and organizations depending on ancient Node.js versions could upgrade or use a vendor that provides this service.

If you are looking for additional proof points, here are a few examples of vulnerabilities that older versions of Node.js are impacted:

- https://nvd.nist.gov/vuln/detail/CVE-2025-23167 affects 18, 16, 14 (llhttp) - medium
- https://nvd.nist.gov/vuln/detail/CVE-2023-5678 affects 16, 14 (openssl) - medium
- https://nvd.nist.gov/vuln/detail/CVE-2024-22019 affects 16, 14 (llhttp) - high

- https://nvd.nist.gov/vuln/detail/CVE-2021-39135 affects 14 (npm) - high

This affects a staggering number of projects. Based on download statistics, Node.js v18, the most recent End-of-Life version, still accounts for approximately 50 million monthly downloads, while earlier legacy versions (v16 and below) continue to see tens of millions of downloads per month. That represents countless applications running on known vulnerable, unsupported runtime environments.

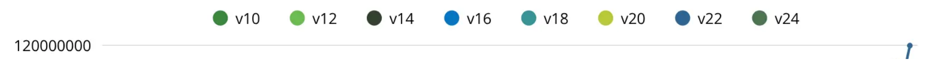
You can check if your Node.js installation is vulnerable to known security vulnerabilities using the is-my-node-vulnerable package. This tool checks your Node.js version against a database of known vulnerabilities and provides guidance on whether you need to upgrade.

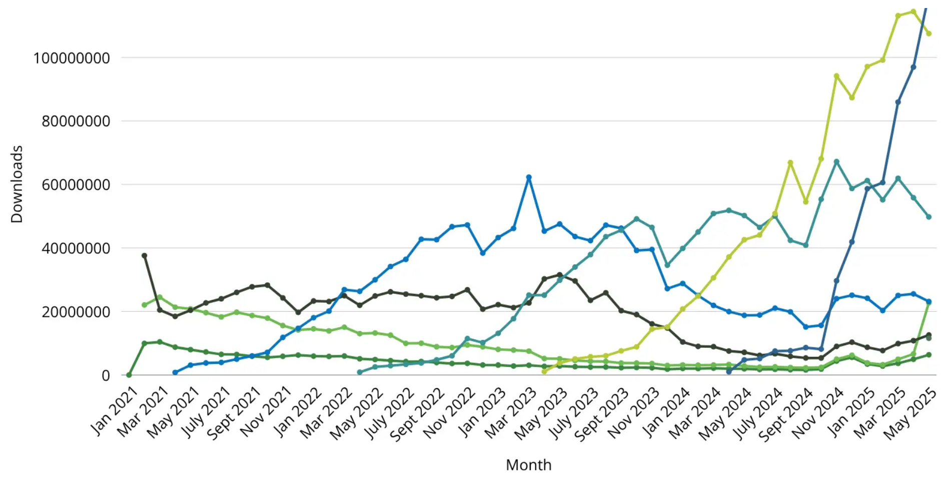
Node.js v22: The Smart Long-Term Choice

While Node.js 20 is currently the maintenance LTS release, Node.js 22 is the smarter upgrade target. Here's why you should skip 20 and go straight to 22:

- Longer Support Window: Node.js 22 is in active LTS status and will be actively supported until April 2027—a full year longer than 20's support window.
- Future-Proofing: By upgrading to 22 now, you avoid another major upgrade cycle in just a couple of years. This saves significant engineering time and reduces upgrade fatigue.
- New Features: 22 offers all the latest features of Node.js, including native TypeScript support (behind a flag).
- Latest Performance Gains: 22 includes all the improvements from 20, plus additional optimizations, offering the best performance available.

The Migration Numbers Game





The download statistics reveal an interesting migration pattern. While the total Node.js downloads have grown to over 350 million monthly downloads across all versions, the distribution shows:

- Node.js v22+: Growing rapidly as teams adopt the future-forward approach, with 120 million downloads.
- Node.js v20: ~100 million monthly downloads (solid but shorter lifespan)
- End-of-Life versions (v18 and below): ~120+ million montlhy downloads (critical security risk)

This means that roughly 30% of the Node.js community is still running on unsupported versions. Rather than making incremental upgrades, smart teams are leapfrogging directly to v22 for maximum future-proofing, or adopting a commercial solution.

Why Skip v20 and Go Straight to v22?

The conventional wisdom might suggest upgrading incrementally to Node.js v20 first, but this is a strategic mistake. Here's why v22 is the better target:

Maintenance Window:

- Node.js v20 LTS: October 2023 - April 2026 (1 year remaining)
- Node.js v22 LTS: October 2024 - April 2027 (2 years remaining)

Upgrade Fatigue Prevention: Major Node.js upgrades require

Upgrade Fatigue Prevention: Major Node.js upgrades require testing, dependency updates, and potential code changes. By going to v22 now, you avoid another upgrade cycle in 2026-2027.

Making the Jump

For Development Teams: Start by auditing your current Node.js usage. Check `node --version` across all your projects and environments. Create a migration timeline that targets 22 directly, skipping the 20 stepping stone.

For DevOps Teams: Update your CI/CD pipelines, Docker images, and deployment scripts to target Node.js 22. Test thoroughly in staging environments, but don't waste time on 20 as an intermediate step.

For Open Source Maintainers: Consider requiring Node.js 22 as your minimum version for new major releases. This positions your project at the forefront of the ecosystem and provides the longest support runway.

Can't Upgrade Right Away? Commercial Support is Available

We understand that some organizations face constraints that prevent immediate upgrades, such as legacy codebases, compliance requirements, or complex dependency chains. If your company cannot upgrade immediately but needs continued security support for Node.js v18 or earlier versions, commercial support is available through HeroDevs.

As part of the OpenJS Ecosystem Sustainability Program partnership, HeroDevs provides Never-Ending Support (NES) for Node.js versions past their official maintenance phase. This includes security patches, compliance assistance, and technical support to help bridge the gap while you plan your upgrade strategy.

However, this should be viewed as a temporary solution—the goal should always be to upgrade to actively supported versions like Node.js 22.

The Bottom Line

With hundreds of millions of monthly downloads across the Node.js ecosystem, the migration to 22 represents a strategic opportunity to future-proof your applications. The security implications alone make upgrading from unsupported versions critical, but the choice between 20 and 22 is about smart long-term planning.

The path forward is clear: Node.js 22 offers the longest support window, best performance, and maximum future-proofing. Don't waste time on incremental upgrades—make the jump directly to 22 and secure your applications for years to come.

Your applications, your users, and your future self will thank you for making the strategic move to Node.js v22 today.

Prev
In Memory of Mikeal Rogers: ...

Next
Making Node.js Downloads R...