Mend.io

Whitepaper

ROI

of Automated Dependency Management with Mend Renovate Enterprise

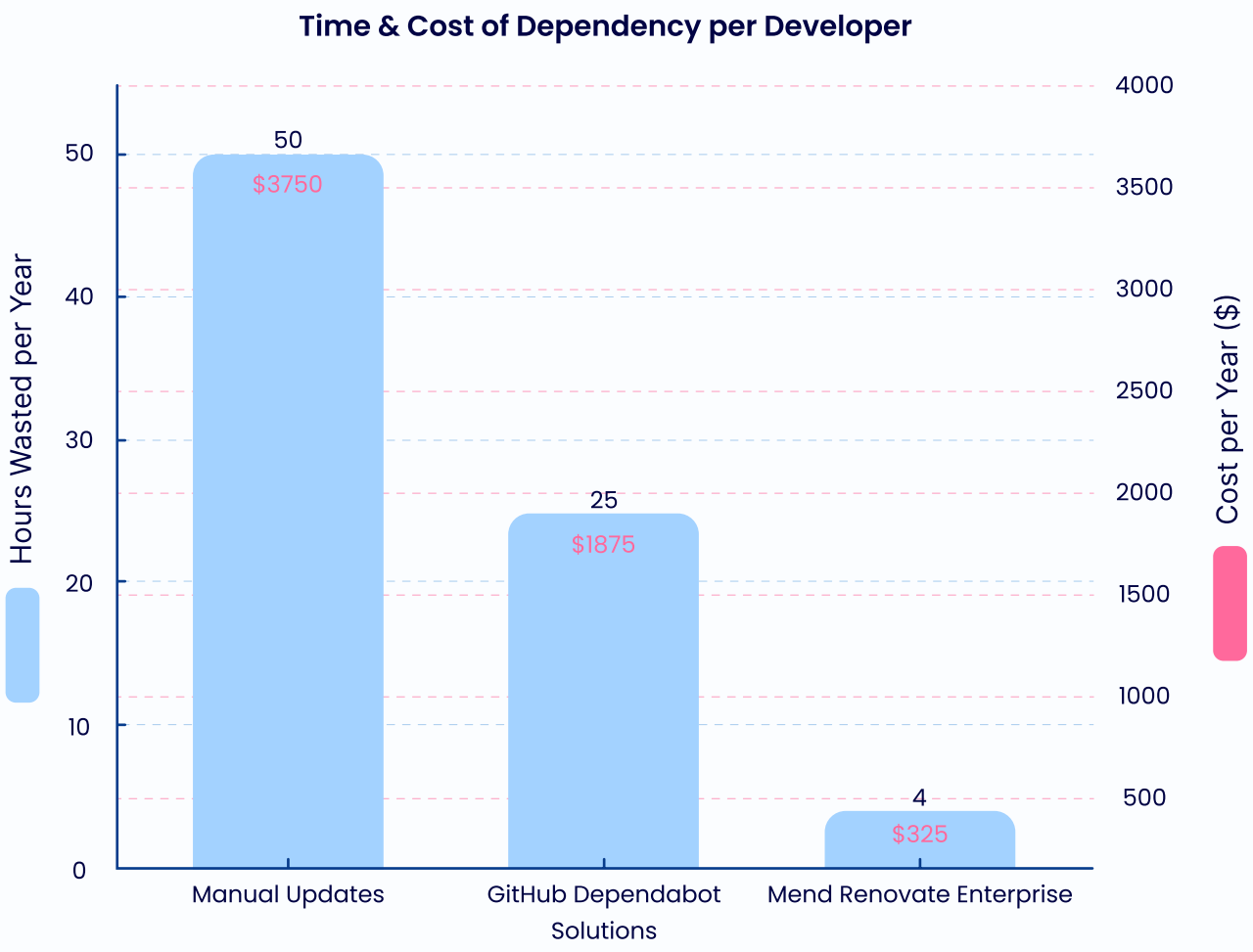
Executive Summary

Dependency management may seem like a small chore, but across thousands of engineers it adds up to millions of dollars in lost productivity. Even modest improvements in automation translate into huge ROI.

Mend Renovate Enterprise minimizes the developer effort needed for dependency updates, freeing engineers to focus on delivering features and reducing security risk. The savings are clear on a per-developer basis and scale with your organization's size.

Per-Developer ROI

Before diving into the calculations, let's note where the salary assumptions come from. We use a $75/hour fully loaded developer cost, approximately equivalent to a $150,000 per year salary, which is a widely accepted industry benchmark. This figure is consistent with data from sources like the Stack Overflow Developer Survey, Glassdoor, and analyst frameworks such as Forrester's Total Economic Impact studies. This standard cost baseline is widely recognized and aligns with how many organizations calculate engineering costs, making it a solid foundation for ROI comparisons.



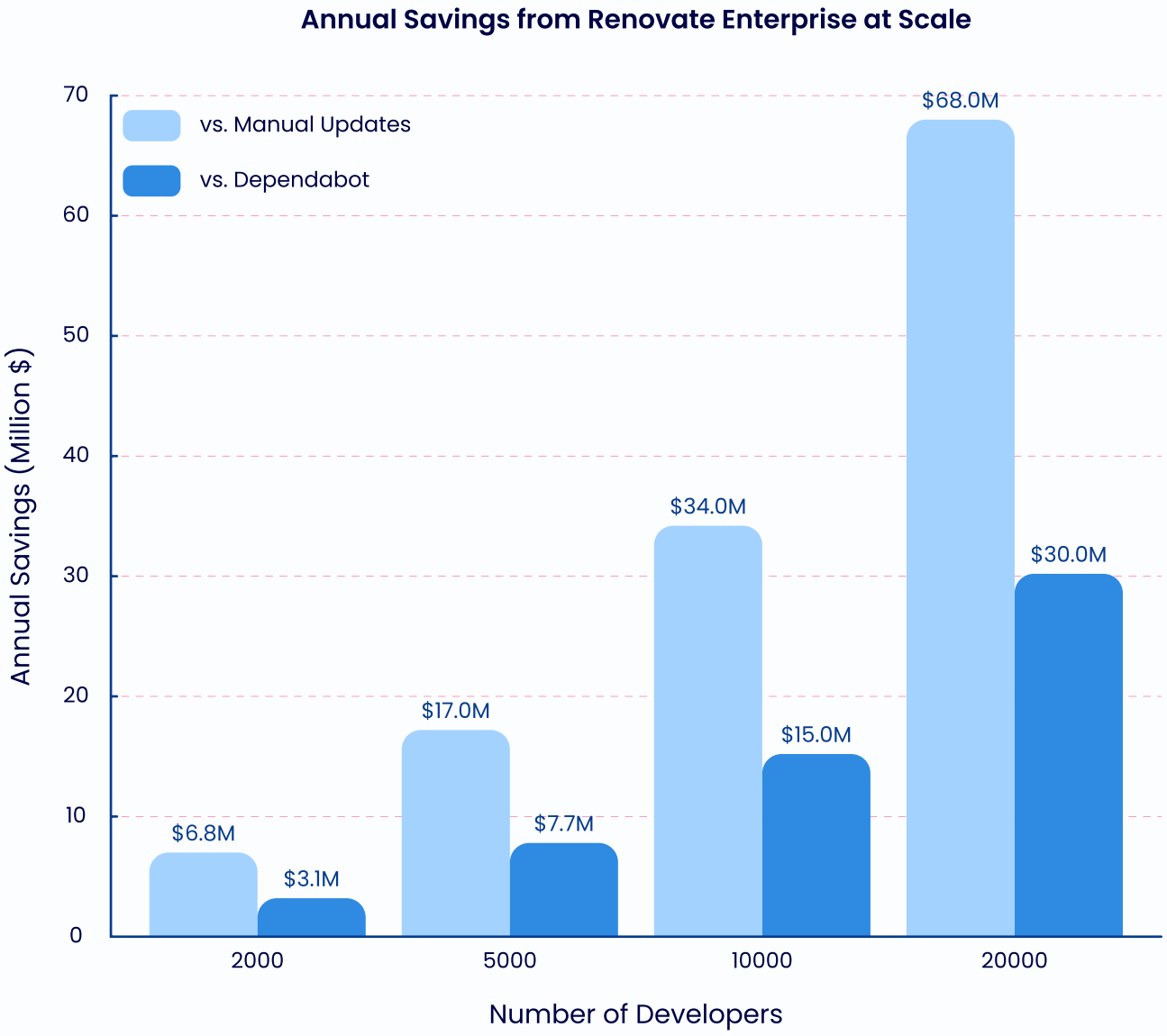
Net annual savings per developer:

$3,425
vs. manual updates

$1,550
vs. Dependabot

Scaling to Enterprise Sizes

When you apply this per-developer math across real-world engineering teams, the results are striking:



Why These Savings Are Real

Time Spent on Manual Maintenance

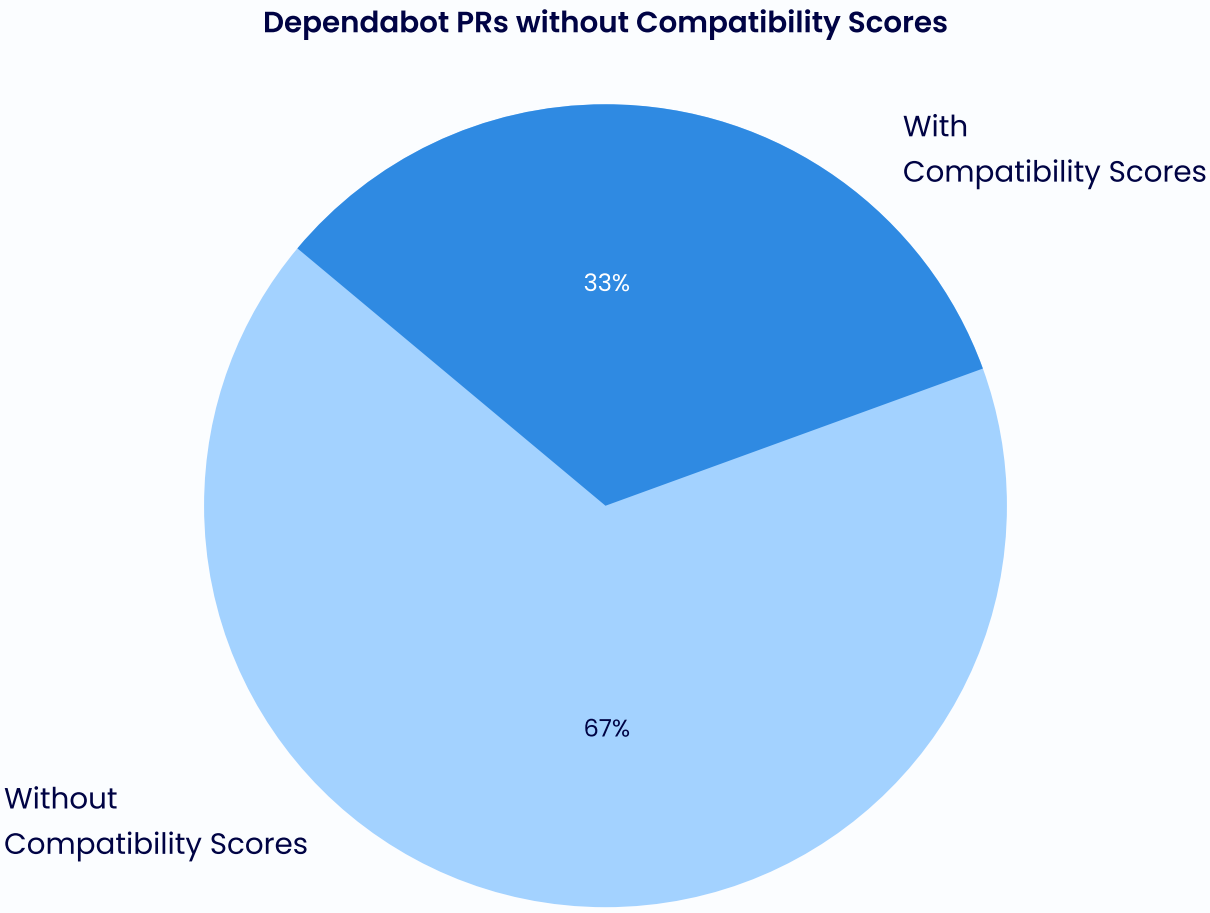
Industry studies and surveys show that developers typically spend at least an hour per week on routine maintenance tasks. This includes activities like checking for new package versions, updating files, testing builds, and troubleshooting issues. Narrowing that down, dependency updates alone easily account for ~1 hour per week on average across teams. While some weeks may be lighter and others much heavier (e.g., a Log4j-style incident), this 1-hour baseline is considered a conservative but realistic average. Multiplied across thousands of developers, even this modest figure compounds to millions of dollars in annual cost.

Dependabot's Partial Automation

GitHub's Dependabot helps reduce this burden by automatically generating pull requests for updates. However, academic research has found that a majority of Dependabot PRs lack compatibility scores, leaving developers unsure if an update will break their build. This means engineers still need to manually verify and test most changes. In practice, Dependabot may cut the workload roughly in half — saving the initial legwork but still leaving review and verification overhead.

Evidence from Automation Studies

Forrester's Total Economic Impact (TEI) studies report that organizations using automated tools like Dependabot or similar saw >75% reductions in remediation time for code maintenance tasks. This aligns with Mend's own claim of ~90% time savings for dependency updates: Renovate extends automation further by grouping PRs, scoring merge safety, and enabling confident auto-merges. This alignment between Mend's ROI model and respected analyst benchmarks strongly suggests the savings projections are reasonable and consistent with independent findings.



A Deeper Dive into Our Model and Assumptions

Credibility of the ROI Model and Supporting Evidence

Mend's ROI framework draws on multiple independent sources to validate its assumptions about time and cost savings:

- **Academic Research on Dependabot:** A peer-reviewed study (Rombaut et al., 2024) analyzed over 579,000 Dependabot pull requests. Roughly two-thirds lacked compatibility score badges, leaving developers unsure if updates were safe. Many developers described these PRs as "irrelevant or hard to trust." This evidence supports the assumption that Dependabot, while useful, still leaves substantial manual verification work.
- **GitHub Documentation & Limits:** GitHub's own documentation highlights Dependabot's limitations, including incomplete coverage of transitive dependencies and limited grouping of updates. Large enterprises may face a flood of individual PRs, whereas Mend Renovate can intelligently bundle and prioritize them. These product differences are well-documented and explain why Renovate delivers superior efficiency.
- **Forrester TEI Study (2022):** Forrester's Total Economic Impact study on GitHub Advanced Security found that automation reduced remediation work by 75–90%. One director reported an 80-hour task dropping to just 6 hours. The composite organization achieved a 433% ROI over three years, with over $140M in developer productivity gains. These results align with Mend's claim of ~90% time savings.
- **Industry Salary Surveys:** The assumption of $150K annual fully loaded developer cost ($75/hour) is supported by sources like Stack Overflow and Glassdoor, and is consistent with Forrester TEI inputs. This makes the cost baseline both conservative and credible.

Validation of Time and Cost Savings Claims

- **Manual updates: ~1 hour per week per developer is a cautious baseline, given the volume of dependencies in modern applications and the spikes caused by incidents like Log4j.**
- **Dependabot: Cutting effort in half (to ~30 minutes) reflects its strengths in proposing updates, balanced by its weaknesses in verification. Research showing two-thirds of PRs without compatibility scores reinforces this.**
- **Mend Renovate: With batching, merge confidence, and scheduling, reducing effort to ~5 minutes per week is ambitious but realistic. Forrester's >90% time reductions in similar tasks make this assumption credible.**

Transparent and Conservative Approach

The model deliberately avoids inflated promises. It uses modest, research-backed assumptions and cites external benchmarks to strengthen credibility. This conservative stance signals that the ROI projections are not hype, but grounded estimates that engineering and financial leaders can trust.

Feature and Capability Comparison: Mend Renovate vs. GitHub Dependabot

A big part of the ROI difference comes from functional advantages that Mend Renovate Enterprise has over GitHub's native Dependabot. These features translate directly into time saved and better outcomes. Table 1 below provides a side-by-side comparison:

Table 1. Feature and Benefits Comparison – Mend Renovate Enterprise vs. GitHub Dependabot (GHAS)

Feature / Capability	Mend Renovate Enterprise	GitHub Dependabot (GHAS)
Transitive Dependency Updates	Full support – Updates both direct and transitive dependencies (beyond just security fixes) across all supported ecosystems.	Partial support – Only updates transitive dependencies in certain cases (e.g., when dependencies in certain cases (e.g., when a security advisory exists); otherwise focuses on direct dependencies.
Pull Request Grouping	Granular control – Can batch multiple updates into one PR based on rules (e.g., all minor updates together, or all updates per project) to minimize PR noise .	Very limited – Can only automatically group some security updates; regular updates are one PR per dependency, which can overwhelm repos with PRs .
Merge Confidence / Compability Scoring	Merge Confidence Workflows: Group, filter, and merge dependency updates based on the confidence of an error-free merge.	Sparse and inconsistent: Only shows a 'compatibility score' if enough projects update the same dependency. ~67% show no score, leaving uncertainty
Centralized Dashboards & Visibility	Org-wide dashboards and reports for complete visibility into update status, dependency management, and policy compliance.	No cross-repo dashboard; monitoring is repo-specific, making update tracking and governance harder.
Supported Platforms & Repos	Multi-platform: Works with GitHub, GitLab, Bitbucket, Azure Repos, and more — including monorepos.	GitHub-only: Not usable across hybrid or non-GitHub ecosystems.
Enterprise SLA & Support	Enterprise-grade support: SLAs, dedicated experts, and custom solutions for large orgs.	Limited: Basic GitHub support only; no dedicated dependency management expertise.

Beyond Time Savings

While this ROI model focuses on productivity, Mend Renovate also provides broader strategic value:

Faster Patching and Security Posture

By accelerating patch cycles, Mend Renovate reduces the window of exposure to known vulnerabilities. This improves security posture and helps organizations meet compliance requirements more easily. Faster remediation not only prevents costly breaches but also reduces the stress of emergency updates.

Governance and Visibility

Centralized dashboards give engineering leadership clear visibility into update status across teams and projects. Leaders can quickly identify lagging areas, track progress against security or compliance goals, and report metrics to executives or auditors. This governance layer ensures updates don't slip through the cracks.

Multi-Platform Support

Enterprises often operate across a mix of code hosting platforms: GitHub, GitLab, Bitbucket, Azure Repos, or monorepos. Mend Renovate supports them all, unifying dependency updates under a single solution. This consistency eliminates gaps and reduces the overhead of managing multiple tools.

Operational Resilience and Support

Unlike basic built-in tools, Mend Renovate Enterprise includes enterprise-grade support and SLAs. If large-scale updates cause issues, organizations can rely on Mend's dedicated support rather than diverting internal engineering capacity. This reduces downtime risk and strengthens operational resilience.

Together, these benefits extend Mend Renovate's impact beyond simple time savings, contributing to stronger security, smoother operations, and improved organizational confidence.

Conclusion

Automating dependency updates can generate multi-million-dollar annual savings, even for organizations with just a few thousand developers. The combination of per-developer productivity gains and scalable impact across engineering teams highlights the scale of potential benefits.

With Mend Renovate Enterprise, organizations can achieve 90%+ efficiency gains in dependency management — turning maintenance into a background process and enabling engineers to focus on delivering value.