Automated Patch Tracking: CampusWatch

Introduction

Dependencies are external libraries, frameworks, or other code components that a project relies on to function correctly[1]. Dependency management, on the other hand, is the process and tools used to handle, track, and resolve these dependencies, ensuring they are properly included and versioned in a project. It helps in identifying the security vulnerabilities in the project before reaching production.

Why it matters

Security: Unpatched dependencies often lead to known vulnerabilities being exploited. Knowing which dependencies are used + whether their patches are missing is essential.
Compliance: Many regulatory or security standards require demonstrating that known vulnerabilities/patches are tracked.
Stability & Compatibility: Some patches will break dependencies; others require prior patches. Without tracking, updates can introduce bugs or inconsistencies.
Risk prioritisation: Not all missing patches have equal risk. Tracking lets organizations prioritize urgent patches (e.g. for high-severity vulnerabilities or critical infrastructure) over less critical ones.

Exploring Patch Tracking Tools for CampusWatch

Traditional methods include OS-level updates (via apt or yum), package manager audits (npm audit, pip-audit, safety), and enterprise vulnerability scanners such as OWASP Dependency-Check[2], Snyk, or Trivy. Each provides valuable insights, but the goal was

[1] https://www.mend.io/blog/dependency-management-vs-dependency-updates-whats-the-difference/
[2] https://owasp.org/www-project-dependency-track/

to adopt a solution that integrates seamlessly with our development workflow while minimizing noise.

I implemented **GitHub Dependabot alerts** for the CampusWatch project. Dependabot offers automated pull requests whenever vulnerabilities or new versions are detected across multiple ecosystems (Node.js, Docker, GitHub Actions). This ensures our team is notified of critical patches in real-time without requiring manual scans.

The Dependabot configuration file used for CampusWatch. It manages daily checks for npm dependencies, weekly checks for Docker base images, and weekly updates for GitHub Actions workflows. Labels and reviewers are included to streamline triage and ensure accountability.

By adopting Dependabot, CampusWatch ensures continuous monitoring of security patches while reducing the operational burden of manual tracking. This approach supports our commitment to **security, automation, and resilience** in production systems.