Tracking Dependency Updates & Security: Do Bots Make a Difference?

Eimaan Saqib & Jaffer Iqbal

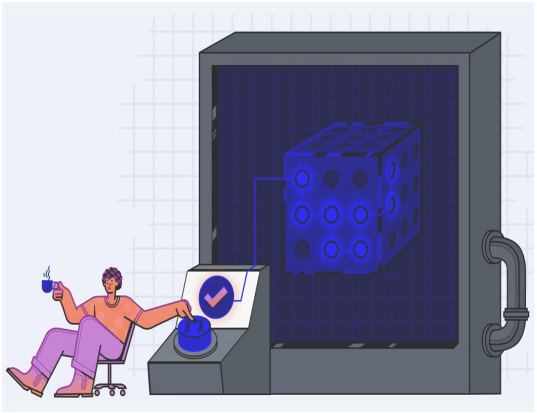
Background

Risks in dependency management

- Security risks
- Technical debt
- Compliance issues

2021 Log4j vulnerability exposed millions of systems

74% open-source codebases contain at least one open-source vulnerability (2024 Synopsys report)



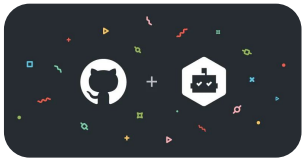
Automated Dependency Management Tools

Automatically open Pull Requests

Update dependencies on a collaborative platform like GitHub

Why?

Dependabot, RenovateBot, SnykBot, Depfu

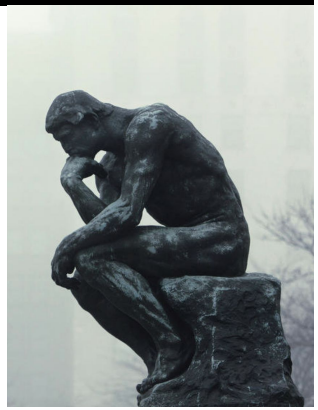


Research Questions

RQ-1: What is the average Dependency Freshness in projects and how does it vary across projects of different sizes, popularity and type?

RQ-2: What are the average Vulnerability Exposure Windows in projects, and how do they vary across projects of different sizes, popularity, and type?

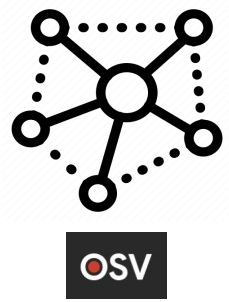
RQ-3: Does the adoption of dependency management bots correlate with reduced Dependency Freshness Vulnerability Exposure Windows?



Dataset & Tools

- **15,117,217 nodes**
- **658,078 artifacts**
- **14,459,139 releases**
- **44,035,495 AddedValue Nodes**
- **Querid via Cypher**
- **77,393 vulnerable releases (1411 artifacts)**
- **197,186 artifacts depending on these directly**
- **GitHub API for metrics and identifying projects**
- **OSV API**

Goblin Framework (represents libraries and releases)



Gathering Project Metrics for Stratified Analysis

- Size: Number of Dependencies + LoC
- Popularity: Number of Dependents + GitHub Stars
- **Project Type**
- Straightforward for numDependencies and numDependents
- **Not so straightforward for the rest!**
- 658,078 artifacts -> reduced set of 38,794 artifacts

Identifying Dependency Freshness

- The time elapsed between the adoption of a dependency by a project and the latest available release of that dependency
- Artifact -> Release -> DependencyVersion (t0) —----> mostRecentVersionOfDependency (t1)
- Aggregate across all releases
- Mean Dependency Freshness + Summary Stats (median+max+min)

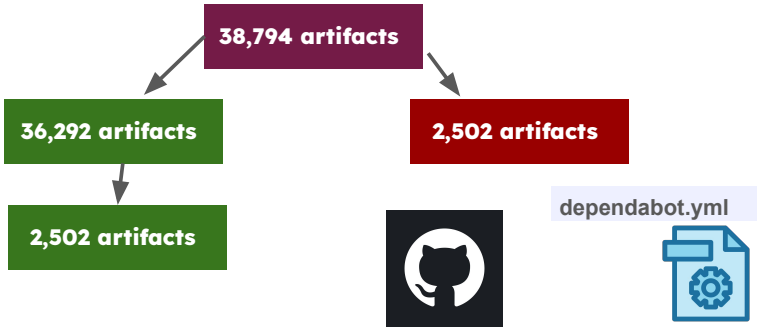
Identifying Vulnerability Exposure Windows

- The windows of time in which an artifact is vulnerable
- ExposureStartTime, ExposureEndTime, cvePublishTime, cveFixTime
- **A ->R -> dep -> B1**
- **A -> R ->dep -> B7 OR A x B**
- No fixes for 29.4% of vulnerable Artifacts (415/1411)

Table 1: Summary of Vulnerability Exposure Windows and Their Descriptions.

Exposure Window Type	Description
True Exposure Window	(*exposureEndTime - exposureStartTime*), quantifies the total duration an artifact remains vulnerable.
Known Exposure Window	(*exposureEndTime - cvePublishTime*), captures the period during which the vulnerability was publicly disclosed yet unaddressed.
Patch Lag Window	(*exposureEndTime* - cveFixTime), reflects the delay between the fix release and the project's adoption of a non-vulnerable version.

Identifying Dependency Freshness and Vulnerability Exposure Windows in Control and Test Group

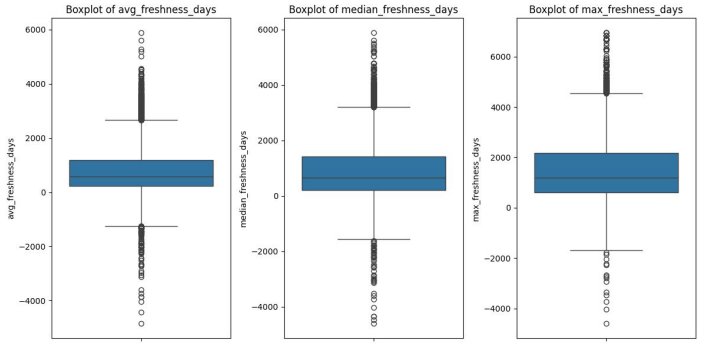


ANALYZING DEPENDABOT IMPACT

- Queried GitHub API for dependabot.yml file
- Control group (36,292) vs treatment group (2502)
- Insufficient representation of bot usage other than Dependabot

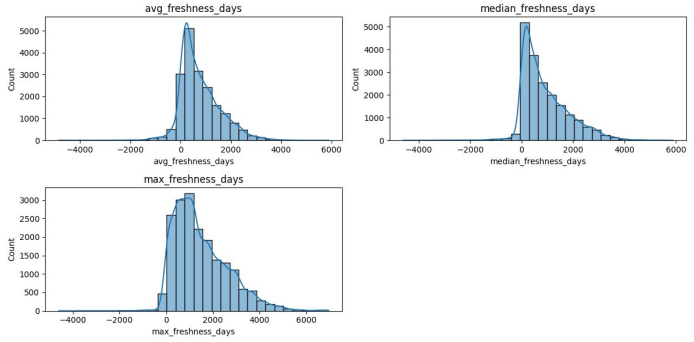
RESULTS

AVERAGE DEPENDENCY FRESHNESS



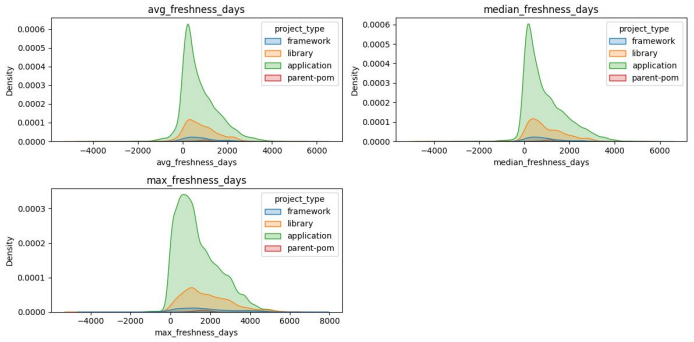
RESULTS

AVERAGE DEPENDENCY FRESHNESS



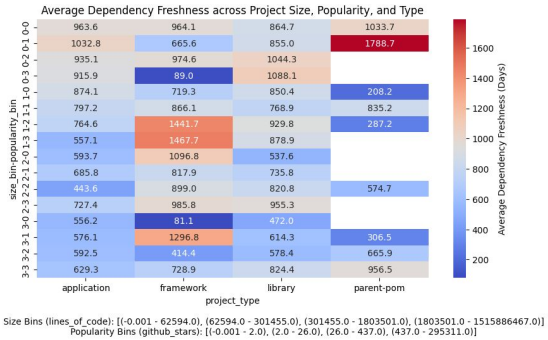
RESULTS

AVERAGE DEPENDENCY FRESHNESS



RESULTS

AVERAGE DEPENDENCY FRESHNESS



Size Bins (lines_of_code): [(-0.001 - 62594.0), (62594.0 - 301455.0), (301455.0 - 1803501.0), (1803501.0 - 1515886467.0)]
Popularity Bins (github_stars): [(-0.001 - 2.0), (2.0 - 26.0), (26.0 - 437.0), (437.0 - 295311.0)]

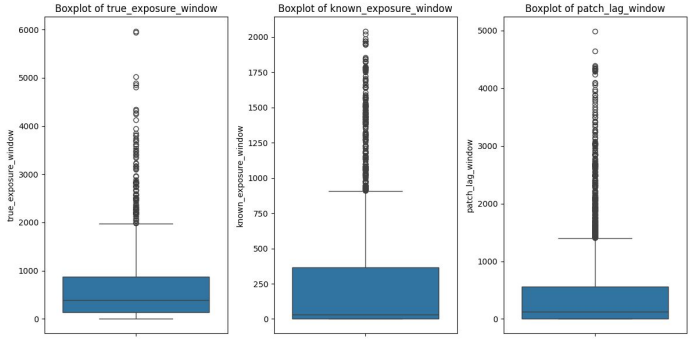
RESULTS

AVERAGE DEPENDENCY FRESHNESS

- Negative correlation for lines of code and number of dependents
- Positive correlation for GitHub stars
- P-values <0.05

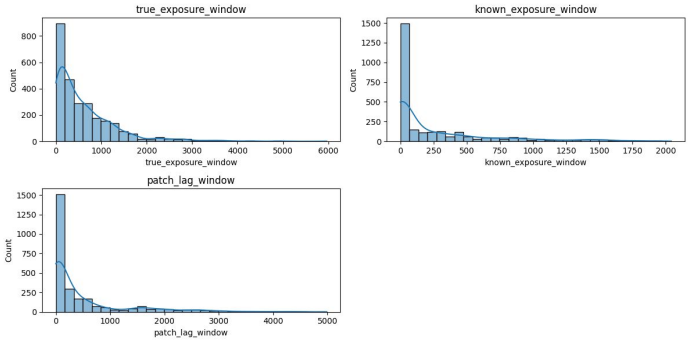
RESULTS

VULNERABILITY EXPOSURE WINDOW



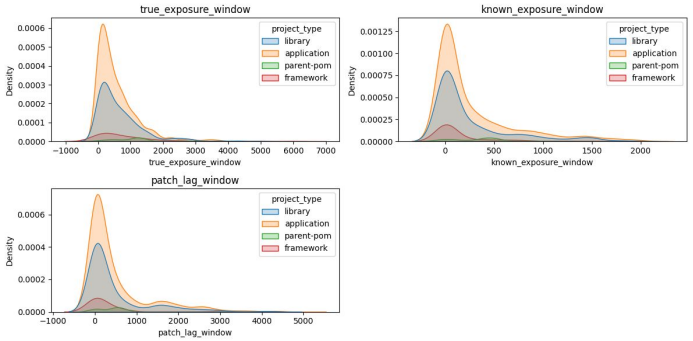
RESULTS

VULNERABILITY EXPOSURE WINDOW



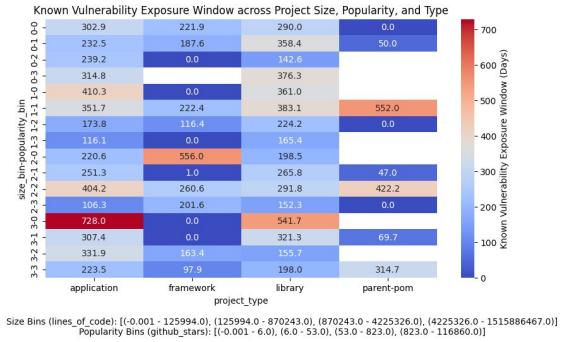
RESULTS

VULNERABILITY EXPOSURE WINDOW



RESULTS

VULNERABILITY EXPOSURE WINDOW



Size Bins (lines_of_code): [(-0.001 - 125994.0), (125994.0 - 870243.0), (870243.0 - 4225326.0), (4225326.0 - 1515886467.0)]
Popularity Bins (github_stars): [(-0.001 - 6.0), (6.0 - 53.0), (53.0 - 823.0), (823.0 - 116860.0)]

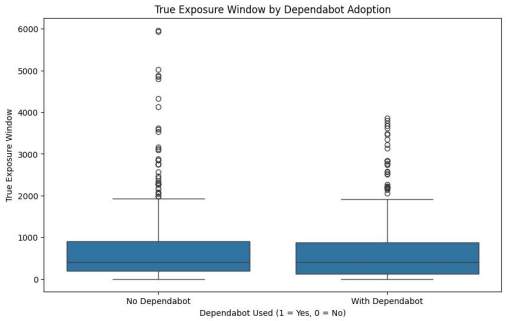
RESULTS

VULNERABILITY EXPOSURE WINDOW

- Negative correlation for number of dependents and GitHub stars
- Positive correlation for lines of code
- P-values <0.05

RESULTS

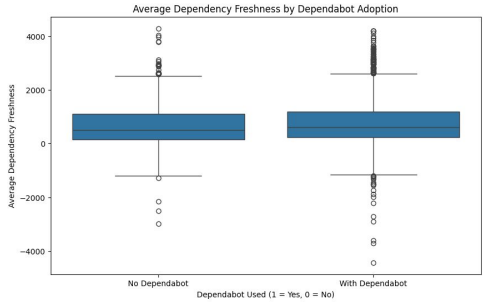
IMPACT OF DEPENDABOT ON VULNERABILITY EXPOSURE WINDOW



- T-statistic: -1.33, P-value: 0.18
- U-statistic: 188886.5, P-value: 0.1655

RESULTS

IMPACT OF DEPENDABOT ON AVERAGE DEPENDENCY FRESHNESS



- T-statistic: 5.93, P-value: 3.3E-9
- U-statistic: 4345206, P-value: 9.47E-9

Threats to Validity

1: Repositories containing a .yml configuration file (like dependabot.yml) may not actively use the dependency management bot	**Verify active bot usage by checking for pull requests authored by the bot, analyzing commit histories for dependency updates, checking config settings**
2: Selection bias could skew results	**Use control groups of similar project size, popularity, and domain.**
3: Causal ambiguity in the findings	**Perform longitudinal study to isolate tool impact.**