Propagation-Based Vulnerability Impact Assessment for Software Supply Chains

ASE 2025

Bonan Ruan, Zhiwei Lin, Jiahao Liu, Chuqi Zhang, Kaihang Ji, Zhenkai Liang
National University of Singapore (NUS)
Contact: bonan.ruan@u.nus.edu

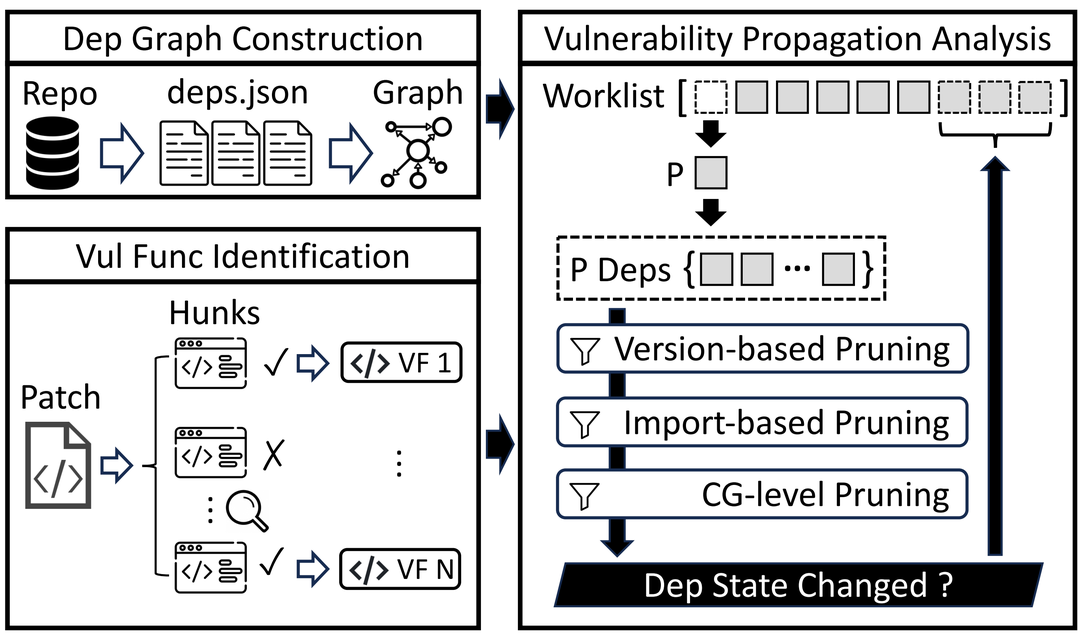
Problem & Motivation

- ***Vulnerability impact in software supply chains is bigger and deeper; we need a method to assess such impact***
- ***Lack of accurate and complete vulnerability propagation analysis:*** Existing works either conduct inaccurate package-level vulnerability propagation analysis, or are callgraph (CG)-level analysis with incomplete consideration of whole-ecosystem dependencies
- ***Lack of metrics for quantifying vulnerability impact across software supply chains:*** Existing metrics (e.g., CVSS, CWE, VPSS) are only used for single vulnerability assessment

Key Ideas

- ***Hierarchical vulnerability propagation analysis algorithm:*** Inspired by data-flow analysis, accurately and efficiently identify affected downstream dependencies across a whole software ecosystem
- ***Time-aware Vulnerability Propagation Scoring System (VPSS):*** make ecosystem-scale impact quantifiable and comparable for software-supply-chain vulnerabilities

Our Approach

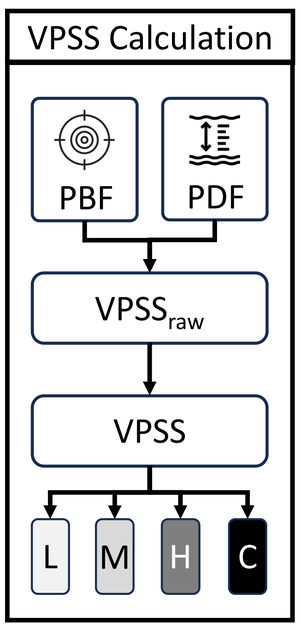


- ***Dep Graph Construction:*** Build a project-level dep graph by analyzing all dep declaration files
- ***Vul Func (VF) Idnetification:*** Generate VF candidates with patch-based method and filter with LLM-assisted strategy
- ***Vul Propagation Analysis:*** Worklist-based algorithm for backward, transitive, CG-level analysis that considers complete dependency scope and project coverage.

  Hierarchical pruning:
  - Dependents on non-vul versions
  - Dependents that declare but do not import upstream
  - Dependents that import but do not call upstream VFs finally

  Propagation continues until a fixed point is reached

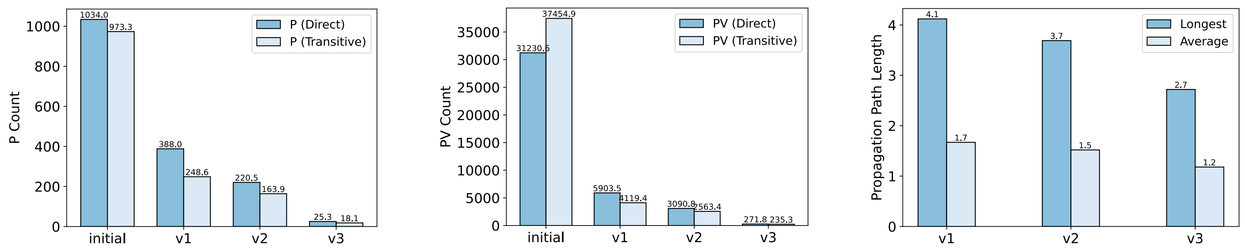
Core Metric: VPSS



- ***Graph Awareness:*** Capture how *widely* (share of affected projects) and *deeply* (dep chain length) a vulnerability propagates
  - ***Propagation Breadth Factor***
  - ***Propagation Depth Factor***
- ***Time Awareness:*** Be dynamic, supporting longitudinal tracking and timely assessment
- ***Compatibility:*** 0–10 scale with 4 severity levels:

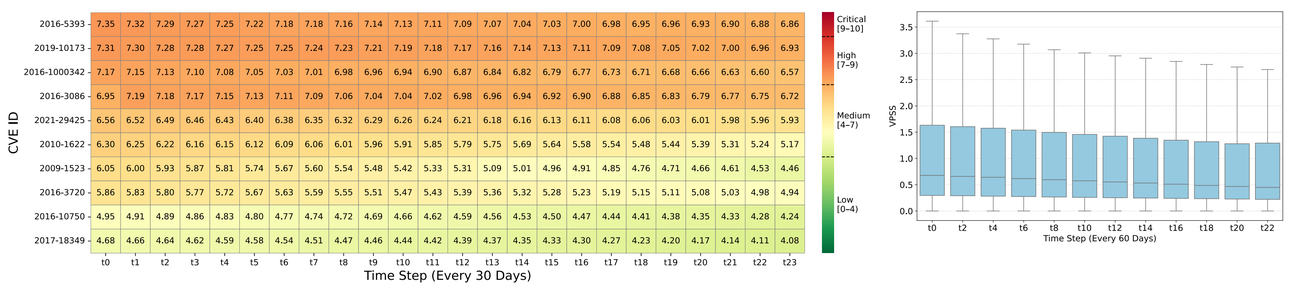
Low [0-4)	Medium [4-7)	High [7-9)	Critical [9-10]

Evaluation (100 Real-World CVEs in Maven)



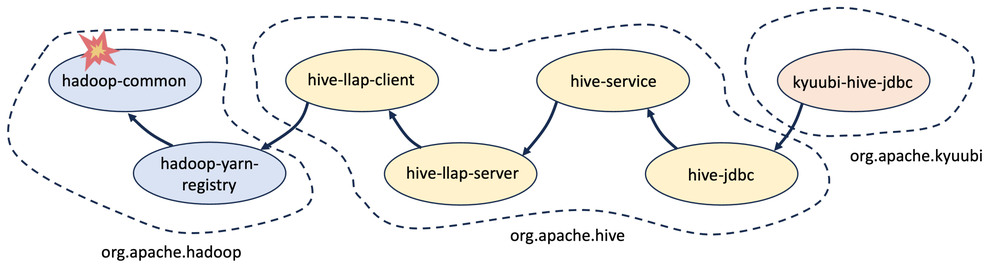
- 97.8% projects & 99.2% project-version releases pruned ↑ scalability
- Longest & avg path lengths ↓ by ≥34.1% & 29.4%
  → massive CG construction avoided

- VPSS trajectories generally decline over time due to patch adoption
- Some CVEs (e.g., CVE-2016-3086) show temporary score increases
- Across the dataset, VPSS scores remain low and gradually stabilize



Case Study: CVE-2016-5393

- Location: *org.apache.hadoop:hadoop-common*
- CVSS score: 8.8 (high severity)
- VPSS score: 7.35 (at disclosure time *t0*), the highest score among all CVEs in our dataset
- The longest propagation chain consists of *7 dependency hops*, spanning critical components of the Hadoop and Hive data processing stacks:



Takeaways

- ***Method:*** We enable the first whole-ecosystem CG-level vul propagation impact assessment
- ***Metric:*** Our VPSS complements CVSS with propagation semantics
- ***Openness:*** Our implementation & dataset are open-sourced

Paper

Code