Improvement Analysis of Function-Level over Package-Level Vulnerability Recommendations

CSE3000
02-07-2021

Niels Mook (Student), under supervision of Mehdi Keshani (PhD Student) and Sebastian Proksch (Assistant Professor)

TU Delft

INTRODUCTION

Programmers reuse each other's code in the form of libraries, packages, **dependencies**, etc.

But what if these dependencies contain **vulnerabilities**?

Dependabot is widely used to alarm you of any vulnerable dependencies on **package-level**.

Having installed Dependabot, it is **spamming** me that most of my dependencies have security issues!! Is my software *that* vulnerable?!

No, most of these warnings are **false positives**, as you probably don't use the vulnerable function of the flagged dependency. That would be **function-level** analysis.

Let's research this recommendation difference in more detail!

MAIN QUESTION

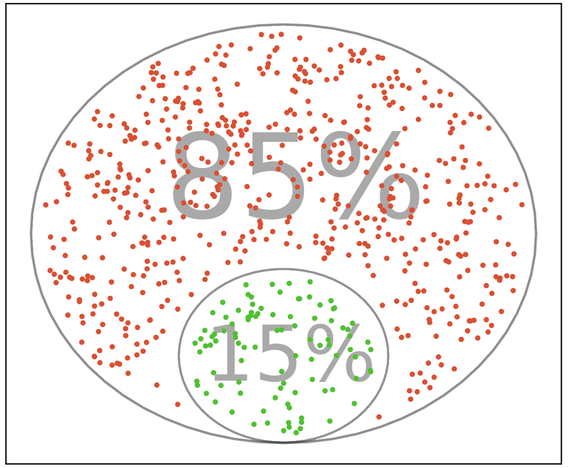
This research aims to provide quantitative insight in the **improvement in recommendation correctness** that fine-grained function-level analysis has over coarse-grained package-level analysis by elimination of false positives.

This is done by generating and comparing the recommendations of both analysis methods for a limited set of repositories.

RESEARCH METHOD

$$\#\,false\ positives = \#\,package\text{-}level\ recommendations - \#\,function\text{-}level\ recommendations$$

- Analysis is done on open-source Java projects hosted on GitHub.
- Vulnerability data (**advisories**) is obtained from the FASTEN Project's database.
- Package-level analysis is performed by analyzing **dependency files** of **dependency managers**.
- Function-level analysis is performed by analyzing **call graphs** through **method tracing**.



Individual vulnerability recommendations in overview

RESULTS

- 7805 repositories starting set
- containing 17,142 Maven POM files
- 259 projects completed both analyses

The following recommendations were generated for these 259 projects:

Vulnerability type	Projects	Recommendations
Package-level	259	680 (100%)
Function-level	78	100 (14.7%)
False positive	239	580 (**85.3%**)

False positives showed an even spread among projects.

CONCLUSIONS

- The elimination of package-level false positives by the fine-grained function-level analysis implementation showed **85.3% correctness improvement**.
- For each used vulnerable function, over 21 internal function calls were at risk, which **emphasizes the improvement.**
- The limited data set cannot represent all repositories well. Research on greater data sets is needed.
- This result shows a first insight in the **significant improvement made by function-level vulnerability** analysis over package-level analysis, promising:
  - Less recommendations to process by developers.
  - More valuable recommendations.

CONTACT DETAILS

Niels Mook
n.l.c.mook@student.tudelft.nl