Am I Breaking My Client's Code? An Analysis of Breaking Changes in the npm Ecosystem

Benjamin Rombaut[1]
[1] *School of Computing, Queen's University, Kingston, Ontario, Canada*
[1] *benjamin.rombaut@queensu.ca*

***Abstract*—The purpose of my project is to examine characteristics of changes in provider packages on npm that cause build failures in client packages. I will collect issue reports generate by Greenkeeper, a dependency management bot, that monitors dependencies of a client package and opens a report when an update to the dependency package breaks a client's build. These issue reports and other associated data can then be analyzed to determine how often providers are breaking their clients builds and what action is needed on the client's part to resolve their build failure.**

I. Problem Statement & Link With Course

Software dependency relationships enables a client package to reuse a certain version of a provider package. Packages in a software ecosystem often release versions containing bug fixes, new functionalities, and security enhancements. However, these updates sometimes include breaking changes, and the provider does not always explicitly follow the *semver* [1] scheme that allows clients to specify updates they would like to receive. Greenkeeper [2] is a tool that owners of a repository can integrate with their project. It sits between npm [3] and GitHub [4], observing all of the modules their repository depends on. Each time one of these dependencies is updated, Greenkeeper opens a new branch with that update. The repository's CI tests run, and Greenkeeper watches them to see whether they pass or fail. Based on the test results and the client's dependency version definitions, Greenkeeper will open up an issue report in the client's repository with information stating which dependency update caused the problem.

My project proposes to examine these issue reports to determine characteristics of changes in provider packages that cause build failures in client packages, as well as what action is needed on the client's part to resolve their build. This proposal is linked to the course in that this information can help release managers avoid or quickly resolve broken builds in their packages.

II. Research Questions & Motivation

The main purpose of my project is to examine characteristics of changes in provider packages that cause build failures in client packages, as well as what action is needed on the client's part to resolve their build failure. Specifically, my research questions are as follows:

RQ1: *How often do provider updates break a client package build?*

RQ2: *How often patch, minor, and major updates of the dependency break a client's build?*

RQ3: *What types of change do client packages need to perform to recover from a build breakage?*

III. Data Set & Analysis

A member of my lab has shared with me a data set he collected using the GitHub API [5]. The data set is based on GitHub projects that have integrated with Greenkeeper. It contains all commits for all projects that have integrated with Greenkeeper, all issue reports for all projects that have integrated with Greenkeeper, all comments on issue reports that have been opened by Greenkeeper, and all events on issue reports that have been opened by Greenkeeper (e.g. tags, updates, etc.).

Using this data set, I can extract the name and version of the provider package that broke the clients build. This information can be used to answer both the first and second research questions. To answer the third research question, the commits for the client's package will have to be examined in order to find the commits that fix the client's build. Once I identify these commits, I can identify further metrics, such as commit size and modified files, to identify types of changes client's make to recover from a build breakage.

IV. Two Related Papers

In 2017, Xavier et al. [6] conducted a large-scale study on API breaking changes in the Java ecosystem. They used a tool named APIDIFF [7] to compare two versions of a library and lists all syntactic changes in public elements. My study takes inspiration from theirs to examine the JavaScript ecosystem.

Cogo et al. [8] found in their study on dependency downgrades in the npm ecosystm that provider packages that break the client's build cause them to downgrade the dependency. However, they did not look in to the types of changes that caused these issues or what other steps clients take to resolve the upgrade issue, other than downgrading the dependency. My project aims to extend their analysis in this regard.

V. Time Planning of Project

My goal is to submit the paper by December 21st so that I can take some time off during the holiday break. The week of October 11th will be spent finalizing my project proposal and giving a presentation to the class. I plan to address any feedback I receive the following week (October 18th) and make adjustments to my proposal as needed.

The week of October 25th will be spent reviewing the data sets that are already available, as well as determining if more data will be required than what was specified in section III to answer the research questions in section II. If more data is required, I will need to write scripts to obtain this data from the GitHub API or NPM registry. The following two weeks (i.e. November 1st and November 8th) will be spent cleaning and pre-processing the data that was collected from the previous weeks.

The weeks of November 15th, November 22nd and 29th will be spent analyzing the cleaned data sets to answer the research questions specified in section II. In addition, I will spend time during this period to prepare my in class project presentation, which will be delivered on December 4th.

The week of December 6th will be spent writing up and finalizing my report for submission. I am keeping the final week (i.e December 13th) before my submission deadline as a buffer, in case any of the previous weekly goals take longer than expected.

References

[1] "Semantic versioning 2.0.0," https://semver.org/, Last accessed: 2020-10-13.

[2] "Greenkeeper - automated dependency management," https://greenkeeper.io/, Last accessed: 2020-10-13.

[3] "npm," https://www.npmjs.com/, Last accessed: 2020-10-13.

[4] "GitHub," https://github.com/, Last accessed: 2020-10-13.

[5] "GitHub REST API," https://docs.github.com/en/free-pro-team@latest/rest, Last accessed: 2020-10-13.

[6] L. Xavier, A. Brito, A. Hora, and M. T. Valente, "Historical and impact analysis of api breaking changes: A large-scale study," in *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2017, pp. 138–147.

[7] A. Brito, L. Xavier, A. Hora, and M. Valente, "Apidiff: Detecting api breaking changes," in *2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. Los Alamitos, CA, USA: IEEE Computer Society, mar 2018, pp. 507–511. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/SANER.2018.8330249

[8] F. R. Cogo, G. A. Oliva, and A. E. Hassan, "An empirical study of dependency downgrades in the npm ecosystem," *IEEE Transactions on Software Engineering*, pp. 1–1, 2019.