Peter Huettl
Garrison Smith
10 February 2018
CS 499 - Project Proposal

NPM Package Dependencies

We are both interested in web technologies and web development, and as such, for our project we will be analyzing the dependency network of modern JavaScript libraries. Specifically, the package management system NPM (Source 1) is an online registry that hosts JavaScript packages. More importantly, NPM manages dependencies between these packages.

So, for example, to include the popular JavaScript package *webpack* (Source 2) into your own project, you would need to also include and install the 22 other packages that *webpack* depends on. Each of these dependencies may also depend on their own selection of packages, and the dependencies expand from there. Although package dependencies are not inherently a bad thing, analyzing this data would provide valuable insight into the interconnectivity and interreliance of popular modern web development technologies.

The network for this project would feature packages as elements, and their dependencies as the relationships between them. Furthermore, there are many questions such a dataset could answer. What are the most popular JavaScript libraries? How are these popular libraries related to other popular libraries? How do package dependencies evolve over time (as the version progresses)? These are just a few examples of the insights to be drawn from such a network.

Problems to be Solved

But, the real problem this dataset could solve is, **how fragile is the modern web development ecosystem?** Or phrased differently, how reliant are some of the most popular JavaScript libraries and websites that use them, on other JavaScript libraries?

The importance of this question becomes clear when you consider that NPM specifically has encountered **two major dependency issues**. On March 22nd, NPM "began observing hundreds of failures per minute, as dependent projects -- and their dependents, and *their* dependents… -- all failed" when a simple 11 line 'left-pad' package was inadvertently deleted from the system (Source 3). These were packages that big corporations depended on, and their websites were broken all due to this over-reliance on packages. On January 6th, a similar incident occurred when a user by the name of 'floatdrop' was accidentally deleted from the NPM registry. As a result, his package 'require-from-string,' which was highly depended on, was also dropped (Source 4). There is also an excellent article outlining a hypothetical XSS attack distributed through NPM (Source 5). So, by analyzing this network, and finding an answer to the proposed problem, our project could provide valuable insight into how web developers use packages.

Data Collection and Analysis

Luckily, seeing as how NPM is a web-focused package management system, they have a public facing web API (Source 6) (Source 7). Although this API does not list all 475,000 NPM packages (Source 1), the repositories, their dependencies, and basic searches can be used to enumerate and collect information regarding the packages. We have already written a crawler script in Python that demonstrates the capability of this enumeration and collection.

Seeing as how we will have to use an algorithm to collect the data in the first place, we will use a sampling method to do this initial collection. Furthermore, after collecting a large enough sample size, we plan on randomly sampling our sampling to generate accurate metrics regarding the connectivity of the network. We also plan on heavily utilizing D3.js and graph visualization tools to intuitively outline any connections we draw during our research. This would be helpful to us while working on the project, and to the final product.

Project Goals

We will be able to evaluate and measure our success towards our goal by creating a metric of which correlates to how reliant a particular package is on other packages. With this metric, we can analyze specific known examples to compare the relative values by hand. Another good metric for success could be how many packages we have properly analyzed and categorized with our network. This metric would ensure that we are striving for accurate and representative network connectivity and results.

By the end of the semester, we plan to have a visually appealing, and thorough analysis of our findings presented both on our website, and in our final write-up. By selecting a refined project scope, we can focus more on presenting meaningful data in an intuitive way using graphs and other visual tools. We can also use our network to visualize a popular modern JavaScript package manager, and apply what we have learned to other, similar package management systems. Systems such as Homebrew, Chocolatey, and Debian Apt are all similar package-based systems and apply to a wide variety of operating systems and devices.

Milestones

- Project Website First Version - February 18th
- Network Data Collected - February 25th
- Project Website Final Version - April 7th
- Project Report - April 7th

Sources

1. NPM - https://www.npmjs.com/
2. Webpack - https://www.npmjs.com/package/webpack
3. left-pad incident - http://blog.npmjs.org/post/141577284765/kik-left-pad-and-npm
4. floatdrop deletion - http://blog.npmjs.org/post/169582189317/incident-report-npm-inc-operations-incident-of
5. NPM XSS Article - https://hackernoon.com/im-harvesting-credit-card-numbers-and-passwords-from-your-site-here-s-how-9a8cb347c5b5
6. NPM API - https://github.com/npm/registry/blob/master/docs/REGISTRY-API.md
7. NPM Download API - https://github.com/npm/download-counts