The design flaw of npm and how ‘everything’ broke it

ISAAC BOAZ, Western Washington University, USA

npm (Node Package Manager) is a popular package manager owned by Microsoft that allows developers to share node packages and libraries. This article will review and analyze the design flaw of npm and how it was broken by a single package. This article will also review how Microsoft responded to the incident and what actions they took to prevent it from happening again.

CCS Concepts: • **Security and privacy** → **Security services**; **Systems security**; • **Social and professional topics** → *Network access restrictions*; **Social engineering attacks**; Universal access; Malware / spyware crime.

Additional Key Words and Phrases: npm, design flaws, everything, left-pad

1 ARTICLE INTRODUCTION

Design flaws in software can sometimes lead to security breaches. More often than not, it is these types of breaches that are the most popularized. However, sometimes a design flaw can lead to a catastrophic failure that was not even intended (or initiated) by a bad actor. With over 1.3 million packages as of April 14th, 2021 [?], npm is the world’s largest software registry [? ?]. As a result, it has been heavily relied upon by developers for years.

Needless to say, the npm registry is a critical piece of infrastructure for the entire JavaScript community (and by extension 99% of websites [?]). Developers expected that the npm registry would be reliable, secure, and complete. However, the left-pad incident in 2016 [?] and the everything package incident in 2023 [?] have shown that the npm registry is not as reliable as developers had hoped.

2 HISTORICAL BACKGROUND

2.1 Introduction

One of the most infamous events in the history of npm was the left-pad incident. Azer Koçulu is a software developer who was working on a personal project called kik [?]. Coincidentally, Kik was also the name of a company in Ontario, Canada [? ?]. The company reached out to Koçulu requesting that he change the name of his project. Koçulu refused, and the company proceeded to file a complaint with npm [?]. npm then decided to transfer the name kik to the company, which caused Koçulu to remove all of his packages from the npm registry, stating

"I want all my modules to be deleted including my account, along with this package. I don’t wanna be a part of NPM anymore. If you don’t do it, let me know how do it quickly. I think I have the right of deleting all my stuff from NPM." [?]

2.2 The left-pad package

As a result of Koçulu’s deciscion, all of his packages (including left-pad) were removed from the npm registry. Despite being only 17 lines of code [?], millions of packages relied on left-pad [?]. As a result, many packages that directly or indirectly relied on left-pad broke, including React, Babel, and Atom.

After an attempt to publish left-pad under a new version failed (due to some major dependencies relying on the old version specificially, which was no longer available), npm resulted to restoring the package from a backup [?]. The total downtime taken to restore the backup was 2.5 hours [?].

Author’s address: Isaac Boaz, boazi@wwu.edu, isaac.k.boaz@gmail.com, Western Washington University, Seattle, Washington, USA.

With the first ever unprecedented un-unpublishing of a package, npm was able to restore the left-pad package and the packages that relied on it.

2.3 Mitigation and Prevention

Once the left-pad incident was resolved, npm published a blog post detailing the incident and the steps they planned to take to prevent it from happening again [?].

'We will make it harder to un-publish a version of a package if doing so would break other packages. We are still fleshing out the technical details of how this will work. Like any registry change, we will of course take our time to consider and implement it with care.' [?]

npm ended up enacting two policies relating to unpublishing packages. The first policy applied on packages that were less than 72 hours old, allowing unpublishing as long as 'no other packages in the npm Public Registry depend on your package' [?].

The second policy applied to packages that were older than 72 hours, allowing unpublishing as long as three conditions were met [?]:

(1) no other packages in the npm Public Registry depend on it
(2) it had less than 300 downloads over the last week
(3) it has a single owner/maintainer

With these new policies in place, npm hoped to prevent a similar incident from happening again. However, these new policies lead towards a new design flaw.

3 THE EVERYTHING PACKAGE

3.1 Everything's Creation

In 2023, a developer by the name PatrickJS decided to work on a package called 'everything'. He teamed up with Evan Boehs, another developer to work on creating an npm package that would declare every single package in the npm registry as a dependency. This invovled working around some obscure npm limitations, such as the maximum number of dependencies being around 800 [?], and the maxiumum package size being 10 MB [?].

In order to work around these limitations, PatrickJS and Evan Boehs decided to chunk their dependencies into smaller packages @everything-registry/chunk-0 through to @everything-registry/chunk-4. At the time there was approximately 2.5 million packages in the npm registry [?], so sub-chunking was also required.

Unsurprisingly, npm also has a rate limit on how quickly packages can be published. Thus, PatrickJS and Evan Boehs used a GitHub Action workflow to publish the packages [?].

3.2 The Aftermath

Upon publishing of the 'everything' package in Dec 2023 [?], PatrickJS and Evan Boehs had (unintentionally) disabled unpublishing every package ever on the npm registry [?]. This was an unforseen consequence of the new unpublishing policies that npm had put in place after the left-pad incident. Namely, *every package was now a dependency of the 'everything' package*, and thus, could not be unpublushed, regardless of age. everything itself could not be removed, as it had one dependency: everything-else [?].

This was also caused by the fact that the 'everything' package depended on a '*' version of all of its packages, which meant that it matched any version of each of its packages.

3.3 Historic Related Incidents

npm was not a strange to packages and projects before 'everything'.

3.3.1 no-one-left-behind. was an npm package that was created in 2018 that similarly depended on 1,000 npm packages [?]. npm managed to remove the package before it caused significant damage [?]. It was then re-uploaded under a different name, this time with 33,000 dependencies [?].

3.3.2 hoarders. was a package that was created near the early conception of npm (2012) that also depended on 11,000 packages at its inception [?]. Isaac Schluter (the founder of npm) talked with the creator of the package and convinced him to unpublish it [?]. A few years later in Aug 19, 2021

REFERENCES

[] Ahmad Nassri. So long, and thanks for all the packages!, April 2020. URL https://blog.npmjs.org/post/615388323067854848/so-long-and-thanks-for-all-the-packages.

[] Luke Karrys and Edward Thomson. About npm, October 2023. URL https://docs.npmjs.com/about-npm.

[] w3schools. What is npm?, January 2023. URL https://www.w3schools.com/whatis/whatis_npm.asp.

[] w3techs. Usage statistics of javascript as client-side programming language on websites, February 2024. URL https://w3techs.com/technologies/details/cp-javascript.

[] Isaac Schlueter. Kik, left-pad, and npm, March 2016. URL https://blog.npmjs.org/post/141577284765/kik-left-pad-and-npm.

[] Theo Browne. How one command broke npm, January 2024. URL https://youtu.be/IzqtWTMFv9Y.

[] Keith Collins. How one programmer broke the internet by deleting a tiny piece of code, March 2016. URL https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code/.

[] Crunchbase. Kik - crunchbase company profile & funding, February 2024. URL https://www.crunchbase.com/organization/kik-interactive.

[] Mike Roberts. A discussion about the breaking of the internet, March 2016. URL https://web.archive.org/web/20160324000044/https://medium.com/@mproberts/a-discussion-about-the-breaking-of-the-internet-3d4d2a83aa4d#.rv5x9r23t.

[] Azer Koçulu. left-pad, August 2014. URL https://github.com/left-pad/left-pad/blob/192444bbe670b3e429164ba0a5808e05f7ca5af4/index.js.

[] Azer Koçulu et al. left-pad, May 2019. URL https://github.com/left-pad/left-pad/network/dependents.

[] npm. npm unpublish policy, January 2023. URL https://docs.npmjs.com/policies/unpublish.

[] PatrickJS. everything - npm, December 2023. URL https://www.npmjs.com/package/everything.

[] Alex Gee. everything-else, June 2015. URL https://www.npmjs.com/package/everything-else.

[] Pierre Krafft. no-one-left-behind, February 2018. URL https://web.archive.org/web/20210512200558/https://www.npmjs.com/package/no-one-left-behind.

[] npm. no-one-left-behind, January 2023. URL https://www.npmjs.com/package/no-one-left-behind.

[] Feross Aboukhadijeh. When "everything" becomes too much: The npm package chaos of 2024, January 2024. URL https://socket.ist/posts/when-everything-becomes-too-much-the-npm-package-chaos-of-2024.

[] Josh Holbrook. hoarders/package.json at f0172e83540c868621d1f67832493ca4f7add3f2 jfhbrook/hoarders, June 2022. URL https://github.com/jfhbrook/hoarders/blob/f0172e83540c868621d1f67832493ca4f7add3f2/package.json.

[] Josh Holbrook. jfhbrook/hoarders: node.js’s most complete "utility grab-bag". dedicated to substack., March 2022. URL https://github.com/jfhbrook/hoarders.

[] Azer Koçulu. I’ve just liberated my modules, March 2016. URL https://web.archive.org/web/20160323000000/https://medium.com/@azerbike/i-ve-just-liberated-my-modules-9045c06be67c#.5g9x9r23t.

A READING RESOURCES

Azer Koçulu himself wrote a blog post about the incident, which can be found at [?].