npm-disputes

Handling Module Name Disputes

This document describes the steps that you should take to resolve module name disputes with other npm publishers. It also describes special steps you should take about names you think infringe your trademarks.

This document is a clarification of the acceptable behavior outlined in the npm Code of Conduct, and nothing in this document should be interpreted to contradict any aspect of the npm Code of Conduct.

TL;DR

1. Get the author email with `npm owner ls <pkgname>`
2. Email the author, CC support@npmjs.com
3. After a few weeks, if there's no resolution, we'll sort it out.

Don't squat on package names. Publish code or move out of the way.

DESCRIPTION

There sometimes arise cases where a user publishes a module, and then later, some other user wants to use that name. Here are some common ways that happens (each of these is based on actual events.)

1. Alice writes a JavaScript module `foo`, which is not node-specific. Alice doesn't use node at all. Yusuf wants to use `foo` in node, so he wraps it in an npm module. Some time later, Alice starts using node, and wants to take over management of her program.
2. Yusuf writes an npm module `foo`, and publishes it. Perhaps much later, Alice finds a bug in `foo`, and fixes it. She sends a pull request to Yusuf, but Yusuf doesn't have the time to deal with it, because he has a new job and a new baby and is focused on his new Erlang project, and kind of not involved with node any more. Alice would like to publish a new `foo`, but can't, because the name is taken.
3. Yusuf writes a 10-line flow-control library, and calls it `foo`, and publishes it to the npm registry. Being a simple little thing, it never really has to be updated. Alice works for Foo Inc, the makers of the critically acclaimed and widely-marketed `foo` JavaScript toolkit framework. They publish it to npm as `foojs`, but people are routinely confused when `npm install foo` is some different thing.

4. Yusuf writes a parser for the widely-known `foo` file format, because he needs it for work. Then, he gets a new job, and never updates the prototype. Later on, Alice writes a much more complete `foo` parser, but can't publish, because Yusuf's `foo` is in the way.

5. `npm owner ls foo` . This will tell Alice the email address of the owner (Yusuf).
6. Alice emails Yusuf, explaining the situation **as respectfully as possible**, and what she would like to do with the module name. She adds the npm support staff support@npmjs.com to the CC list of the email. Mention in the email that Yusuf can run npm owner `add alice foo` to add Alice as an owner of the foo package.
7. After a reasonable amount of time, if Yusuf has not responded, or if Yusuf and Alice can't come to any sort of resolution, email support support@npmjs.com and we'll sort it out. ("Reasonable" is usually at least 4 weeks.)

REASONING

In almost every case so far, the parties involved have been able to reach an amicable resolution without any major intervention. Most people really do want to be reasonable, and are probably not even aware that they're in your way.

Module ecosystems are most vibrant and powerful when they are as self-directed as possible. If an admin one day deletes something you had worked on, then that is going to make most people quite upset, regardless of the justification. When humans solve their problems by talking to other humans with respect, everyone has the chance to end up feeling good about the interaction.

EXCEPTIONS

Some things are not allowed, and will be removed without discussion if they are brought to the attention of the npm registry admins, including but not limited to:

1. Malware (that is, a package designed to exploit or harm the machine on which it is installed).
2. Violations of copyright or licenses (for example, cloning an MIT-licensed program, and then removing or changing the copyright and license statement).
3. Illegal content.
4. "Squatting" on a package name that you plan to use, but aren't actually using. Sorry, I don't care how great the name is, or how perfect a fit it is for the thing that someday might happen. If someone wants to use it today, and you're just taking up space with an empty tarball, you're going to be evicted.
5. Putting empty packages in the registry. Packages must have SOME functionality. It can be silly, but it can't be nothing. (See also: squatting.)

6. Doing weird things with the registry, like using it as your own personal application database or otherwise putting non-packagey things into it.
7. Other things forbidden by the npm Code of Conduct such as hateful language, pornographic content, or harassment.

If you see bad behavior like this, please report it to abuse@npmjs.com right away. **You are never expected to resolve abusive behavior on your own. We are here to help.**

TRADEMARKS

If you think another npm publisher is infringing your trademark, such as by using a confusingly similar package name, email abuse@npmjs.com with a link to the package or user account on https://www.npmjs.com/. Attach a copy of your trademark registration certificate.

If we see that the package's publisher is intentionally misleading others by misusing your registered mark without permission, we will transfer the package name to you. Otherwise, we will contact the package publisher and ask them to clear up any confusion with changes to their package's `README` file or metadata.