Improving the Security of a Major Open Source Project One Step at a Time

Michael Dawson, Rafael Gonzaga, Paula Paul

May 11 2023

Michael Dawson

Node.js lead for Red Hat and IBM

- Active Node.js community member
  - Node.js Collaborator
  - Node.js Technical Steering Committee member
  - Active in a number of Working group(s)
- Active OpenJS Foundation member
  - Voting Cross Project Council Member
  - Community Director 2020-2022

Twitter: @mhdawson1

GitHub: @mhdawson

Linkedin: https://www.linkedin.com/in/michael-dawson-6051282

Rafael Gonzaga

- Staff Engineer at Nearform
- Made in Brazil 🇧🇷

_rafaelgss

RafaelGSS

rafaelgss

Open Source

- **Node.js** Technical Steering Committee (**TSC**) member
- **Node.js Security WG** lead
- **Node.js Releaser**

Paula Paul

- Field CTO at **Nearform**
- ***Standing in, as best I can, for Rafael…***

@paulapaultweets

@paulapaul

NearForm DX Team (3 Node Core Contributors!) and OSPO Technical Sponsor

OpenJS Foundation Board Member

Co-Chair, Grace Hopper Celebration Open Source Day

Open Source fan & Node.js admirer 💙💛

Overview

- Background
    - The Node.js Project
    - OSSF Funding
- Sharing our Experience
    - Reactive - The life of a security vulnerability
    - Proactive - The security working group
- How you can help

The Node.js Project

- Open Open Source
- 3,215 contributors, 96 collaborators
- Widely used
  - >1 Billion downloads from Node.js org last year
  - A top OpenSSF criticality score value
- Security has always been top of mind
- Volunteers are poor match for time critical work

OSSF Funding

- Full time resource
  - starting in 2022
  - continuing in 2023
- Provides “critical mass” to enable community to make good progress

https://openssf.org/

Reactive - The life of a security vulnerability

- Threat model
- Security reports
- Creating fixes
- Security releases

- A real example

Reactive - The life of a security vulnerability

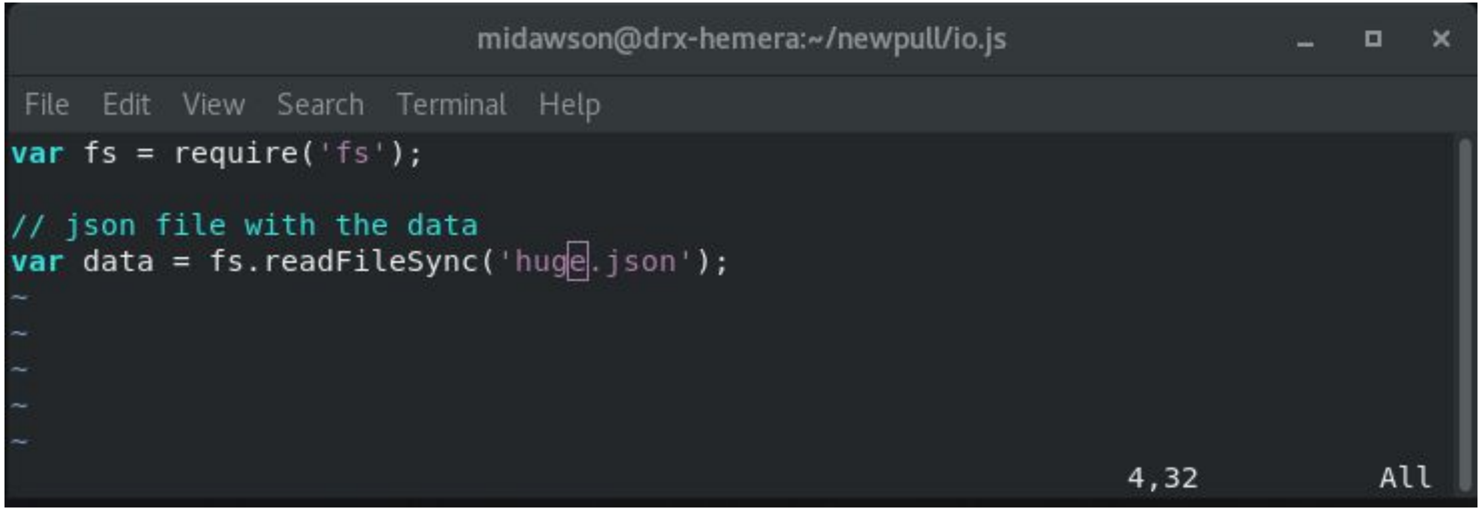
- **Threat model**
- Security reports
- Creating fixes
- Security releases

Threat Model - Our Experience

Without a thread model discussions often feel like:

Image by stockking - on Freepik - https://www.freepik.com/free-photo/young-beautiful-couple-man-women-quarreling-gesturing-having-fight-crazy-frustrated-standing-orange-wall_13055114.htm

Threat Model - examples


midawson@drx-hemera:~/newpull/io.js
File Edit View Search Terminal Help

```
var fs = require('fs');

// json file with the data
var data = fs.readFileSync('huge.json');
~
~
~
~
~
                                                          4,32          All
```

Reactive - The life of a security vulnerability

- Main components
  - What we trust
  - What we don’t trust
  - Examples
- Published in security.md
  - Recent addition last year
  - Hard to define :(

Threat Model - examples

From the Threat Model:

If Node.js loads configuration files or runs code by default (without a specific request from the user), and this is not documented, it is considered a vulnerability. Vulnerabilities related to this case may be fixed by a documentation update.

Example of what is not a vulnerability

External Control of System or Configuration Setting (CWE-15)

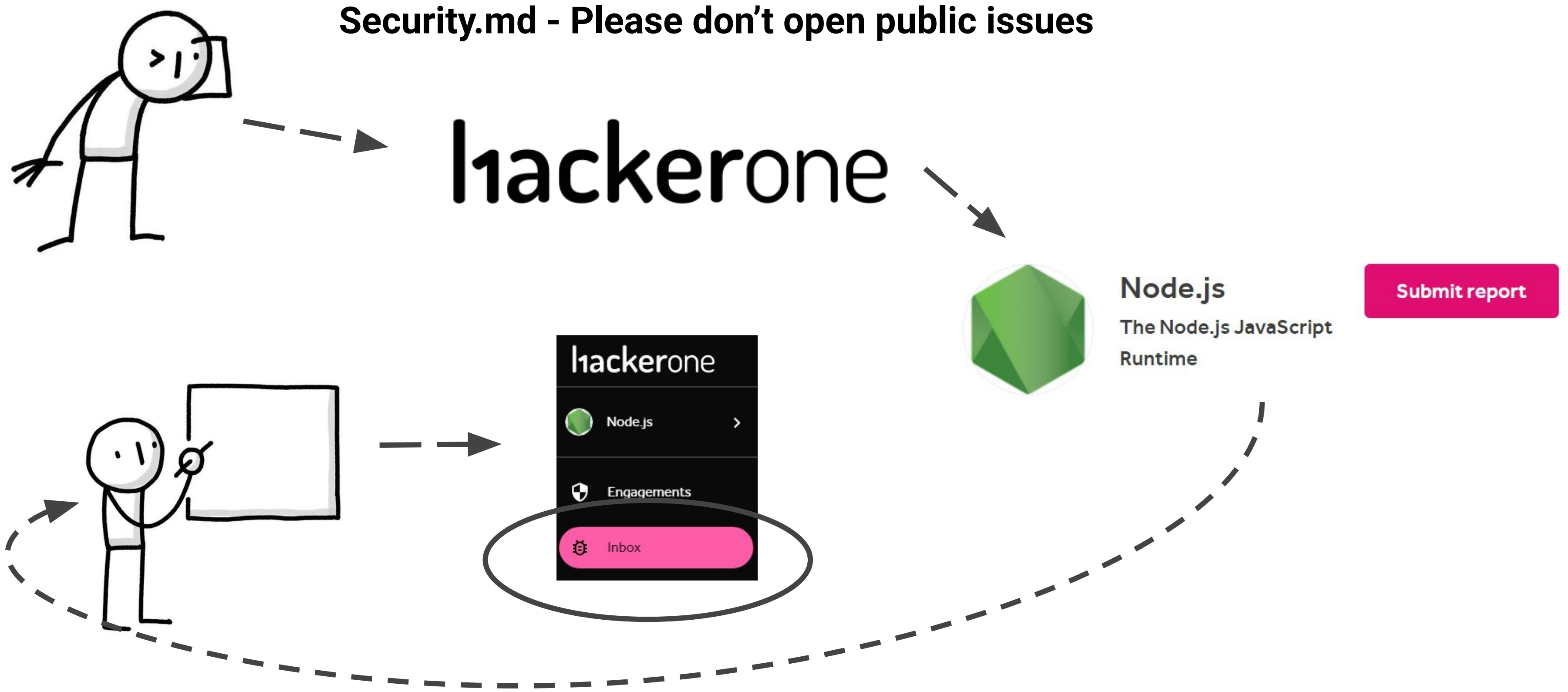
- If Node.js automatically loads a configuration file which is documented no scenario that requires modification of that configuration file is considered a vulnerability.

Reactive - The life of a security vulnerability

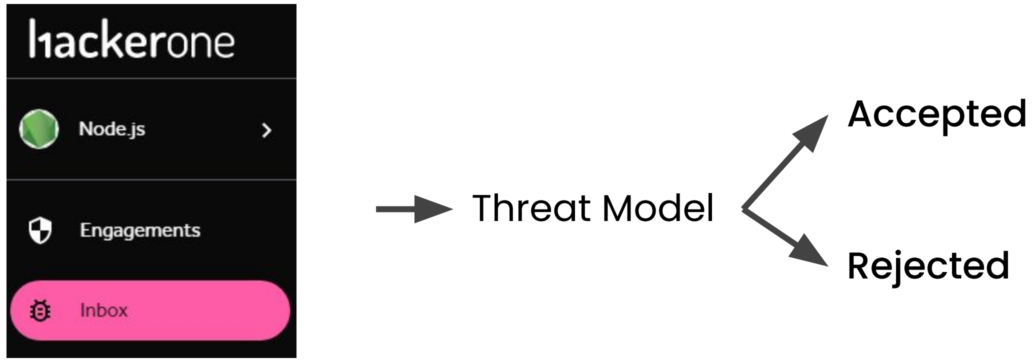
- Threat model
- **Security reports**
- Creating fixes
- Security releases

Security Reports - Submission

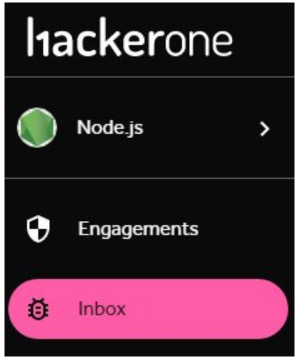
Security.md - Please don't open public issues



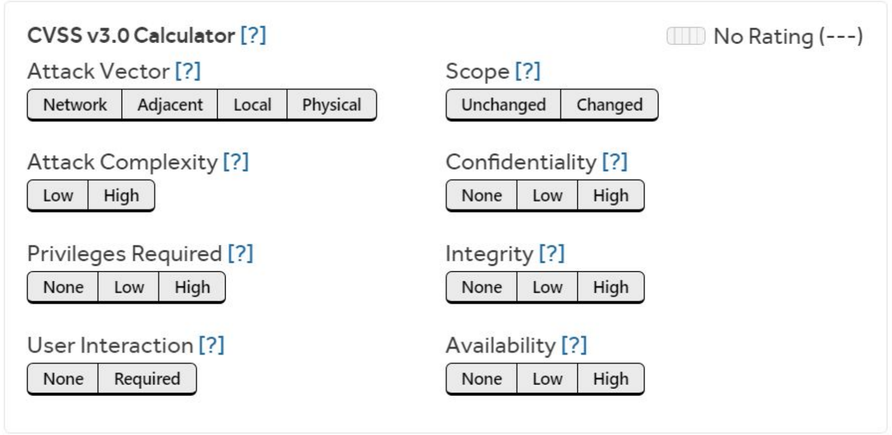
Security Reports - Triage



Security Reports - CVE Assignment

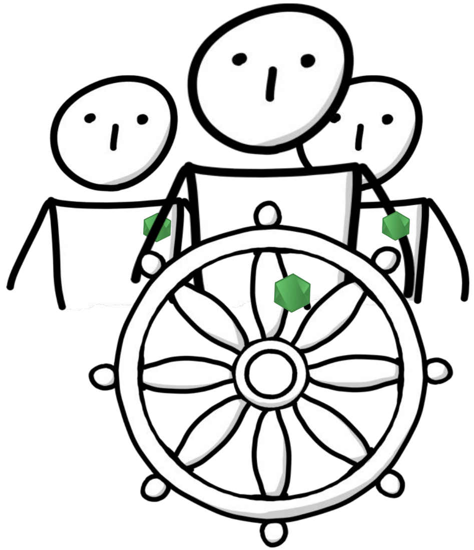


$\rightarrow$ Threat Model $\nearrow$ Accepted $\searrow$



Security Reports - Our experience

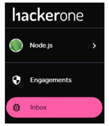
- What did not work
  - Email
  - Ad Hoc triaging
  - Small number of triagers (even if dedicated)
- What's working
  - Triage team > 3 people
  - Triage rotation
  - Hackerone
    - Private place to report
    - Public afterwards
    - Easy CVE assignment



Reactive - The life of a security vulnerability

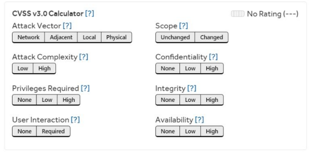
- Threat model
- Security reports
- **Creating fixes**
- Security releases

Creating Fixes



Threat Model

Accepted



CVSS v3.0 Calculator [?] No Rating (---)

Attack Vector [?]: Network | Adjacent | Local | Physical

Scope [?]: Unchanged | Changed

Attack Complexity [?]: Low | High

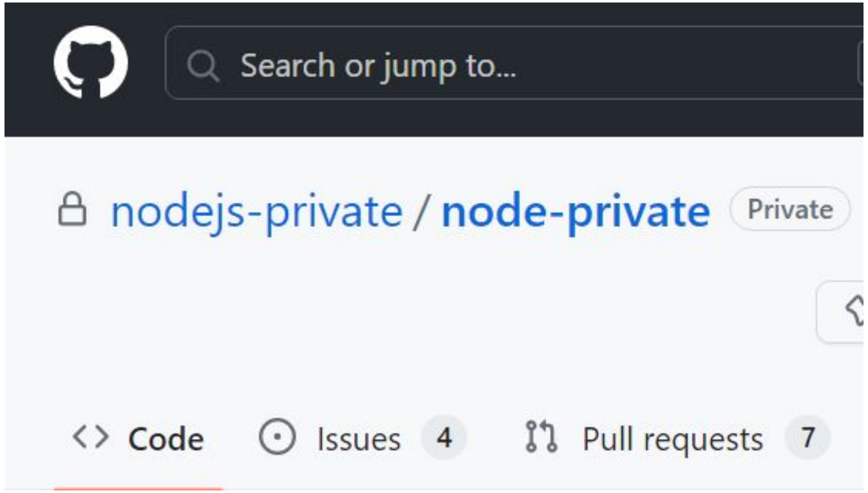
Confidentiality [?]: None | Low | High

Privileges Required [?]: None | Low | High

Integrity [?]: None | Low | High

User Interaction [?]: None | Required

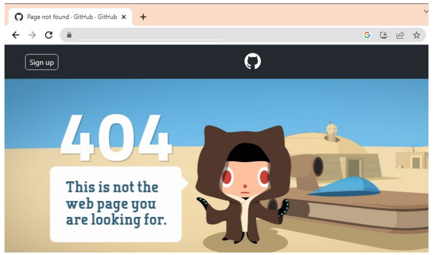
Availability [?]: None | Low | High



Creating Fixes - Our experience

- People availability
  - People with expertise are often busy
    - OSSF funding helped here
  - Often hard to get platform expertise

- Harder to work in private
  - Limited CI/testing
  - Harder to pull in people to help
  - Have lock CI when doing security release


Page not found · GitHub · GitHub
Sign up
404
This is not the web page you are looking for.

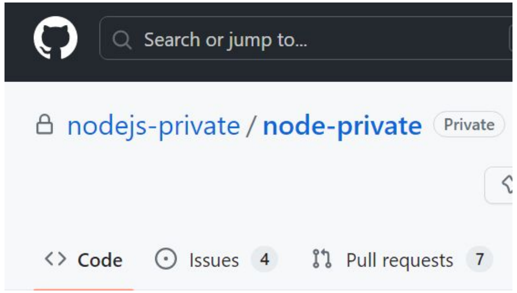
Reactive - The life of a security vulnerability

- Threat model
- Security reports
- Creating fixes
- **Security releases**

Security Releases

- Well documented security release process
- 26 Steps
  - Coordinating many collaborators
  - Advance notice to ecosystem
  - Advance notice to related teams
  - Information about vulnerabilities fixed
  - CI Lock/unlock

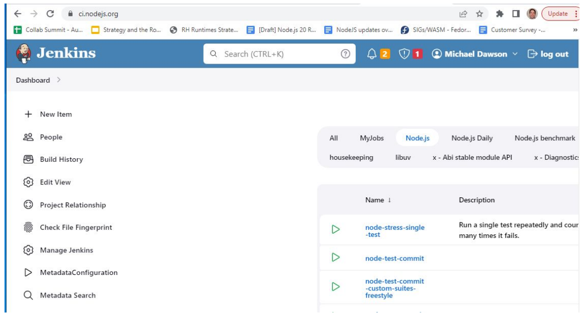
Security Releases



Search or jump to...

nodejs-private / node-private Private

Code Issues 4 Pull requests 7



ci.nodejs.org

Collab Summit - Au... | Strategy and the Ro... | RH Runtimes Strate... | [Draft] Node.js 20 R... | NodeJS updates ov... | SIGs/WASM - Fedor... | Customer Survey -...

Jenkins Search (CTRL+K) 2 1 Michael Dawson log out

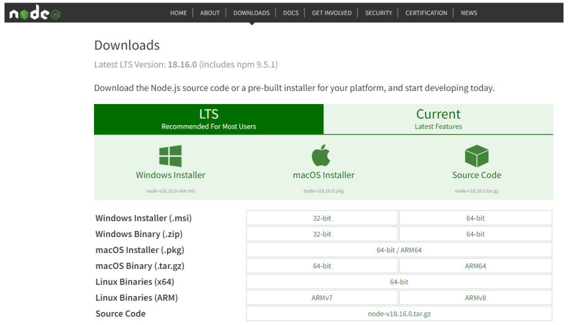
Dashboard >

- New Item
- People
- Build History
- Edit View
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- MetadataConfiguration
- Metadata Search

All | MyJobs | Node.js | Node.js Daily | Node.js benchmark | housekeeping | libuv | x - Abi stable module API | x - Diagnostic

Name ↓	Description
node-stress-single-test	Run a single test repeatedly and cour many times it fails.
node-test-commit	
node-test-commit-custom-suites-freestyle	

Private Release CI



HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Downloads

Latest LTS Version: **18.16.0** (includes npm 9.5.1)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS — Recommended For Most Users | Current — Latest Features

Windows Installer node-v18.16.0-x64.msi | macOS Installer node-v18.16.0.pkg | Source Code node-v18.16.0.tar.gz

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64
Linux Binaries (x64)	64-bit	
Linux Binaries (ARM)	ARMv7	ARMv8
Source Code	node-v18.16.0.tar.gz	

Security Releases - Our experience

- What did not work
  - Releasers doing on their own
  - Ad-hoc coordination
  - Dedicated release steward
- What's working
  - Security release steward rotation
  - >3 security stewards in rotation

Security Releases - release stewards rotation

Release Steward	Organization
Matteo Collina	Platformatic
Michael Dawson	Red Hat
Bryan English	Datadog
Rafael Gonzaga	NearForm
Juan José	NodeSource
Joe Sepi	IBM

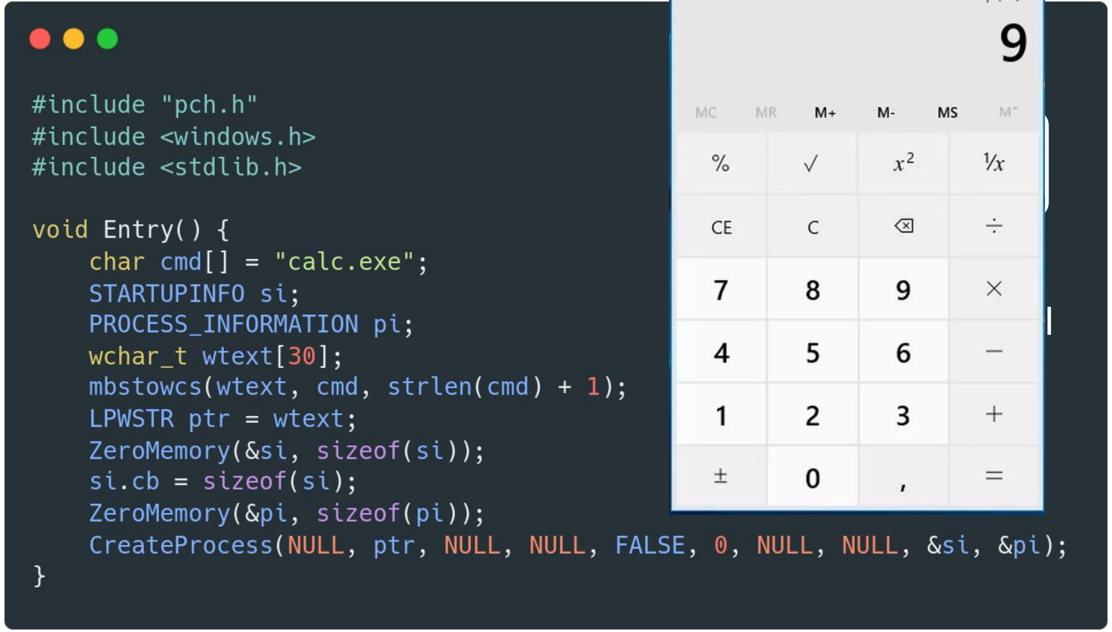
https://github.com/nodejs/node/blob/main/doc/contributing/security-release-process.md#security-release-stewards

Reactive - The life of a security vulnerability

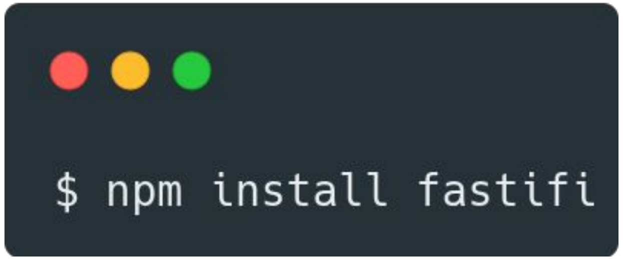
- **A real example**

Windows OpenSSL

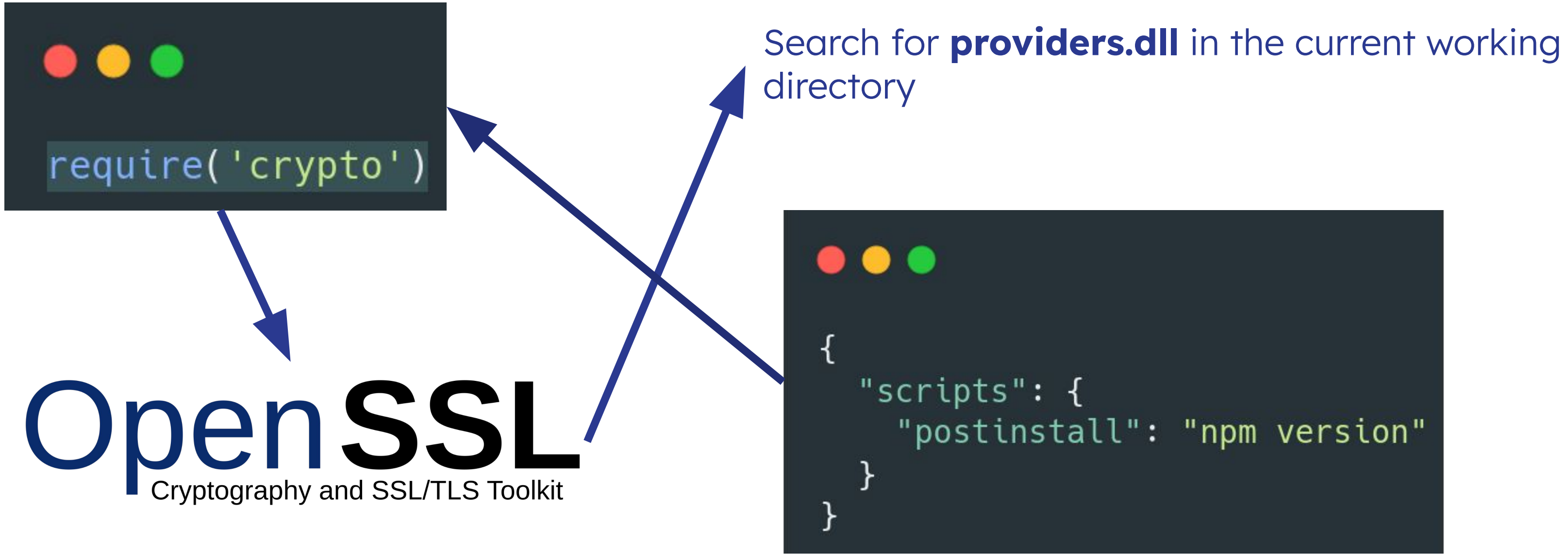
aphy and SSL/TLS Toolkit



```
#include "pch.h"
#include <windows.h>
#include <stdlib.h>

void Entry() {
    char cmd[] = "calc.exe";
    STARTUPINFO si;
    PROCESS_INFORMATION pi;
    wchar_t wtext[30];
    mbstowcs(wtext, cmd, strlen(cmd) + 1);
    LPWSTR ptr = wtext;
    ZeroMemory(&si, sizeof(si));
    si.cb = sizeof(si);
    ZeroMemory(&pi, sizeof(pi));
    CreateProcess(NULL, ptr, NULL, NULL, FALSE, 0, NULL, NULL, &si, &pi);
}
```



```
$ npm install fastifi
```



```
require('crypto')
```

Search for **providers.dll** in the current working directory

```
{
  "scripts": {
    "postinstall": "npm version"
  }
}
```

Proactive - Security Working Group

- History and Active Roster
- Recent Successes
- Current Initiatives
- How to get involved!

Security WG History and Active Roster

Rafael Gonzaga
NearForm

Marco Ippolito
NearForm

Michael Dawson
Red Hat

Ulises Gascon
One Beyond

Thomas Gentilhomme
MyUnisoft

Bradley Farias
SocketSecurity

Ashish Kurmi
StepSecurity

And more... roster in GitHub!

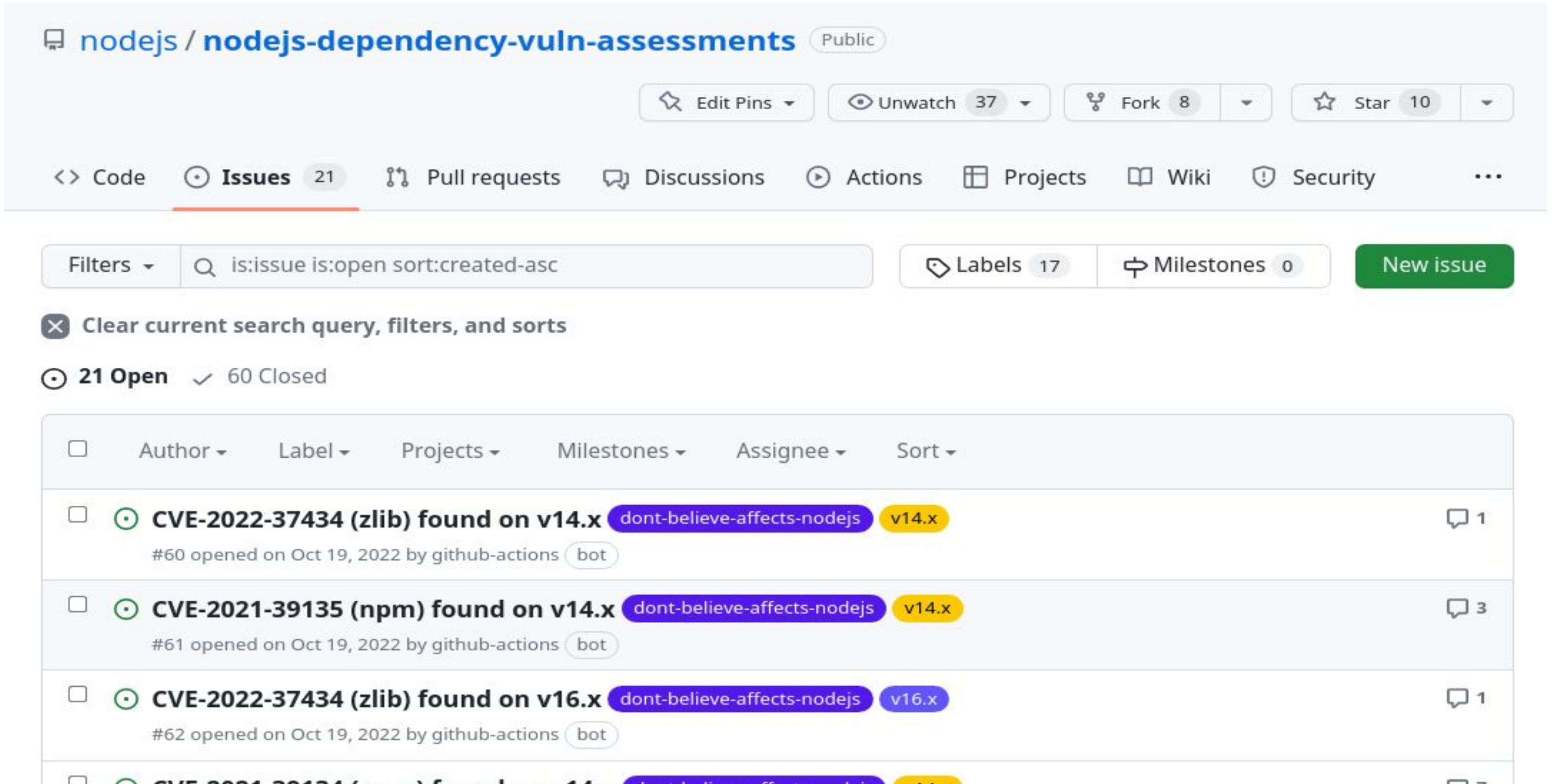
- Node Security Project Vulnerability Database
  - Donated to Node.js Foundation; became out of date
- OSSF funding provided “critical mass” to reform the WG
- Primary focus is now on Node.js itself

Security Working Group - Recent Successes

- Threat Model (covered previously)
- **Dependency Vulnerability Checks**
- Permissions Model
- Security Best Practices

Being Proactive: Dependency Vulnerability Checks

https://github.com/nodejs/nodejs-dependency-vuln-assessments/issues


nodejs / nodejs-dependency-vuln-assessments Public

Edit Pins · Unwatch 37 · Fork 8 · Star 10

Code · Issues 21 · Pull requests · Discussions · Actions · Projects · Wiki · Security

Filters · is:issue is:open sort:created-asc · Labels 17 · Milestones 0 · New issue

Clear current search query, filters, and sorts

21 Open · 60 Closed

Author · Label · Projects · Milestones · Assignee · Sort

CVE-2022-37434 (zlib) found on v14.x dont-believe-affects-nodejs v14.x · 1
#60 opened on Oct 19, 2022 by github-actions bot

CVE-2021-39135 (npm) found on v14.x dont-believe-affects-nodejs v14.x · 3
#61 opened on Oct 19, 2022 by github-actions bot

CVE-2022-37434 (zlib) found on v16.x dont-believe-affects-nodejs v16.x · 1
#62 opened on Oct 19, 2022 by github-actions bot

Security Working Group - Recent Successes

- Threat Model (covered previously)
- Dependency Vulnerability Checks
- **Permissions Model**
- Security Best practices

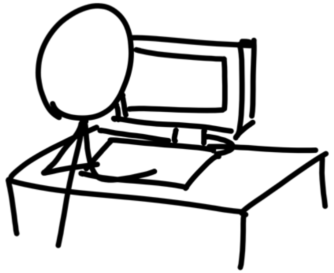
Being Proactive:

Permission Model

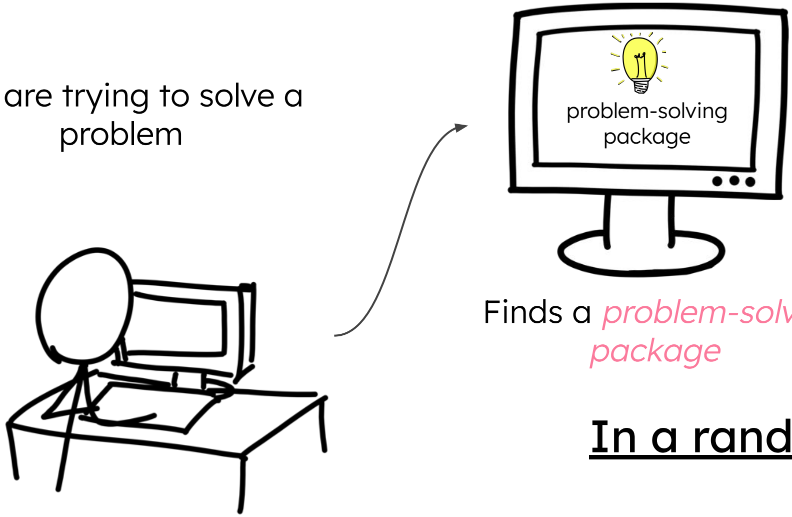
Node.js v20

--experimental-permission

You are trying to solve a problem



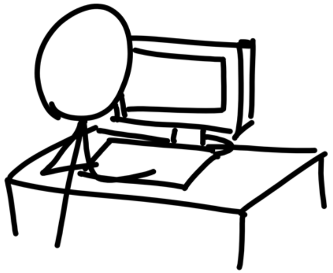
You are trying to solve a problem

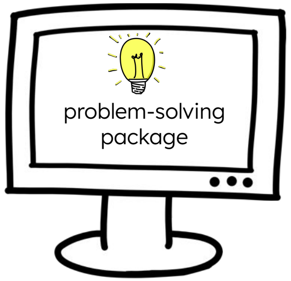


Finds a *problem-solver package*

In a random tutorial...

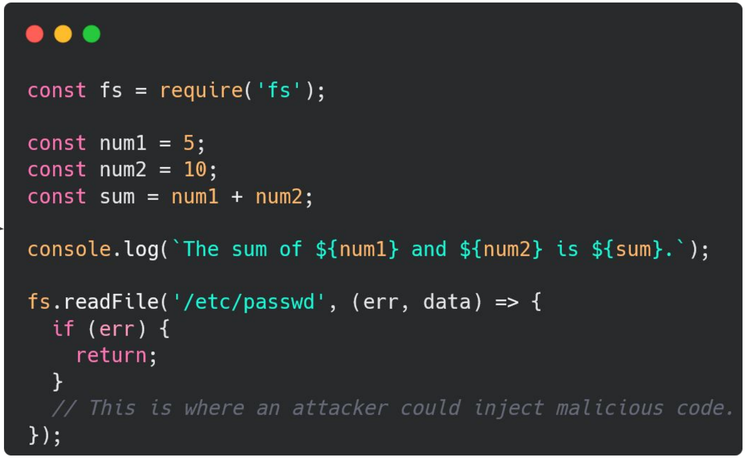
A humble dev trying to solve a problem





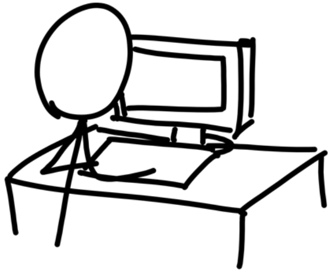
Finds a problem-solver package

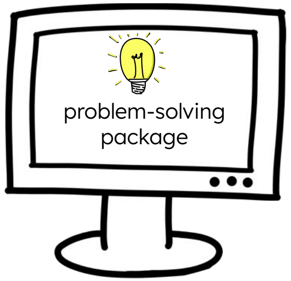
The problem-solver-package looks like this



```
const fs = require('fs');

const num1 = 5;
const num2 = 10;
const sum = num1 + num2;

console.log(`The sum of ${num1} and ${num2} is ${sum}.`);

fs.readFile('/etc/passwd', (err, data) => {
  if (err) {
    return;
  }
  // This is where an attacker could inject malicious code.
});
```

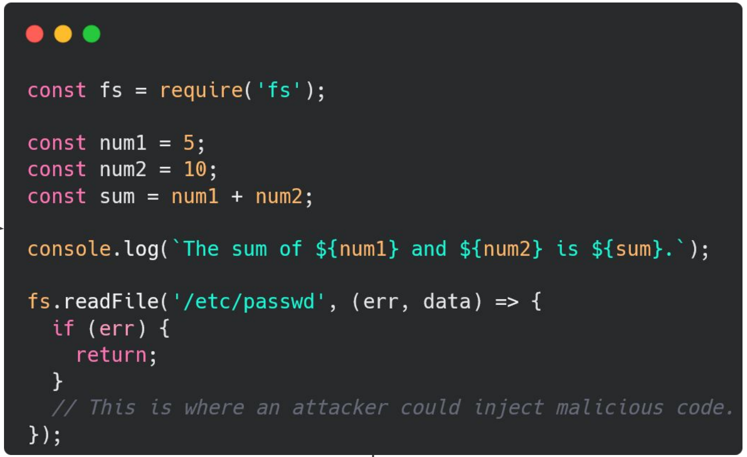
A humble dev trying to solve a problem





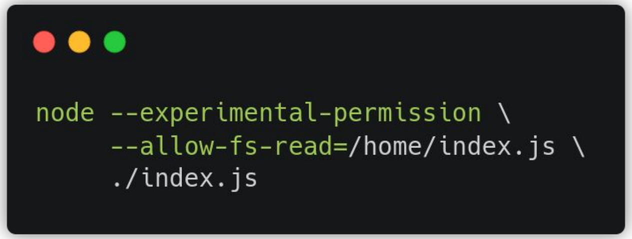
Finds a problem-solver package

The problem-solver-package looks like this



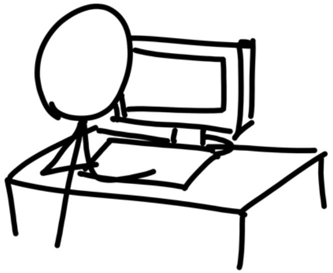
```
const fs = require('fs');

const num1 = 5;
const num2 = 10;
const sum = num1 + num2;

console.log(`The sum of ${num1} and ${num2} is ${sum}.`);

fs.readFile('/etc/passwd', (err, data) => {
  if (err) {
    return;
  }
  // This is where an attacker could inject malicious code.
});
```

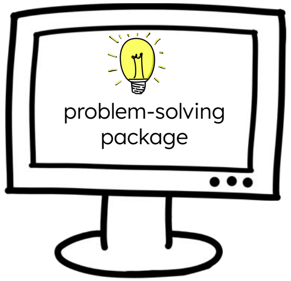
You decide to be cautious and use the permission model



```
node --experimental-permission \
     --allow-fs-read=/home/index.js \
     ./index.js
```

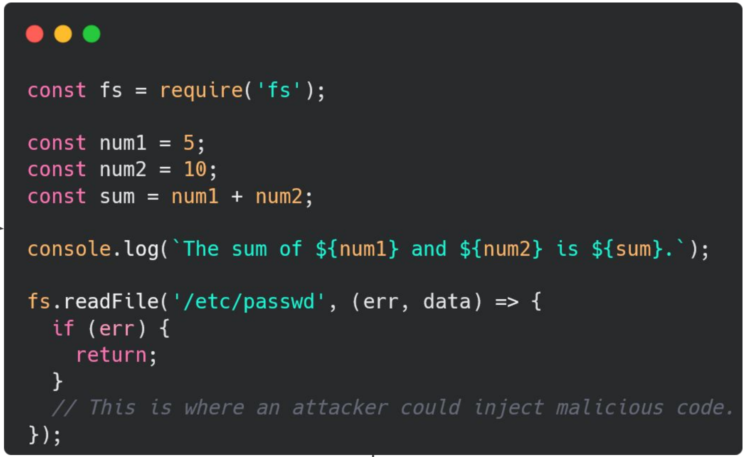
A humble dev trying to solve a problem





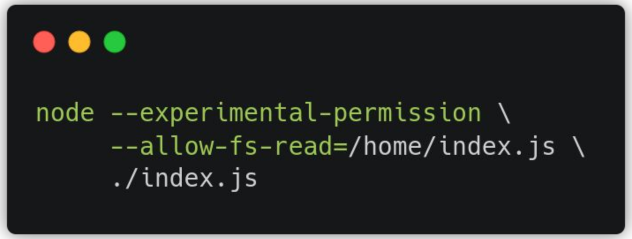
Finds a problem-solver package

The problem-solver-package looks like this



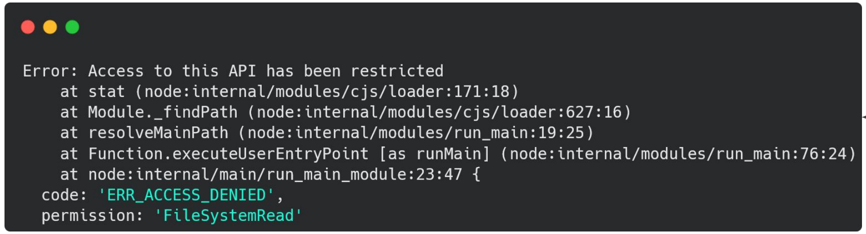
```
const fs = require('fs');

const num1 = 5;
const num2 = 10;
const sum = num1 + num2;

console.log(`The sum of ${num1} and ${num2} is ${sum}.`);

fs.readFile('/etc/passwd', (err, data) => {
  if (err) {
    return;
  }
  // This is where an attacker could inject malicious code.
});
```

You decide to be cautious and use the permission model



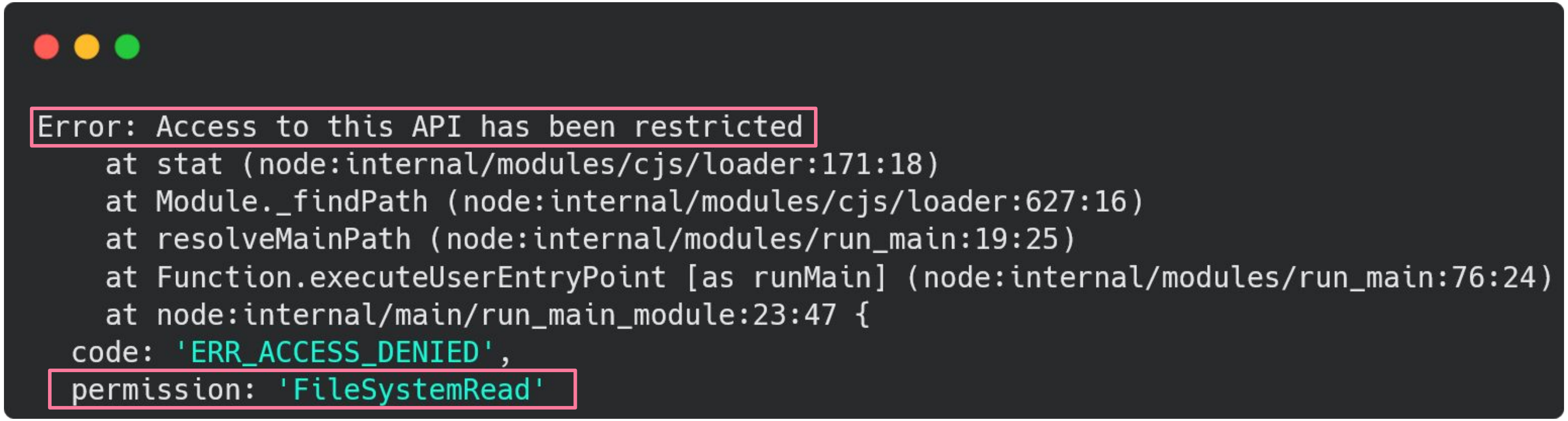
```
node --experimental-permission \
     --allow-fs-read=/home/index.js \
     ./index.js
```

And you get saved by the Permission Model!



```
Error: Access to this API has been restricted
    at stat (node:internal/modules/cjs/loader:171:18)
    at Module._findPath (node:internal/modules/cjs/loader:627:16)
    at resolveMainPath (node:internal/modules/run_main:19:25)
    at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:76:24)
    at node:internal/main/run_main_module:23:47 {
  code: 'ERR_ACCESS_DENIED',
  permission: 'FileSystemRead'
```

Permissions Model



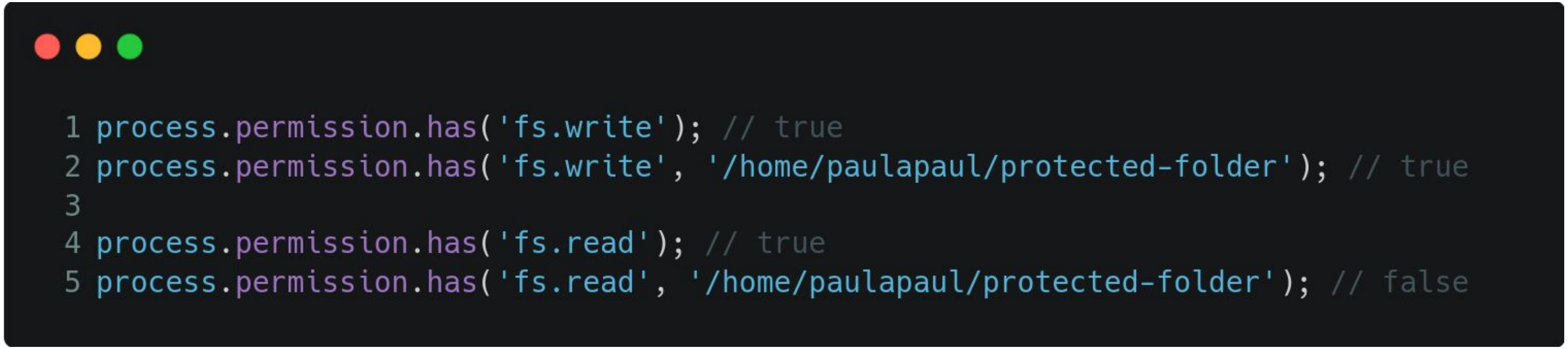
```
Error: Access to this API has been restricted
    at stat (node:internal/modules/cjs/loader:171:18)
    at Module._findPath (node:internal/modules/cjs/loader:627:16)
    at resolveMainPath (node:internal/modules/run_main:19:25)
    at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:76:24)
    at node:internal/main/run_main_module:23:47 {
  code: 'ERR_ACCESS_DENIED',
  permission: 'FileSystemRead'
```

Permissions Model

```
--allow-fs-read
--allow-fs-write
--allow-child-process
--allow-worker
```

Runtime API

- has(scope [,parameters])

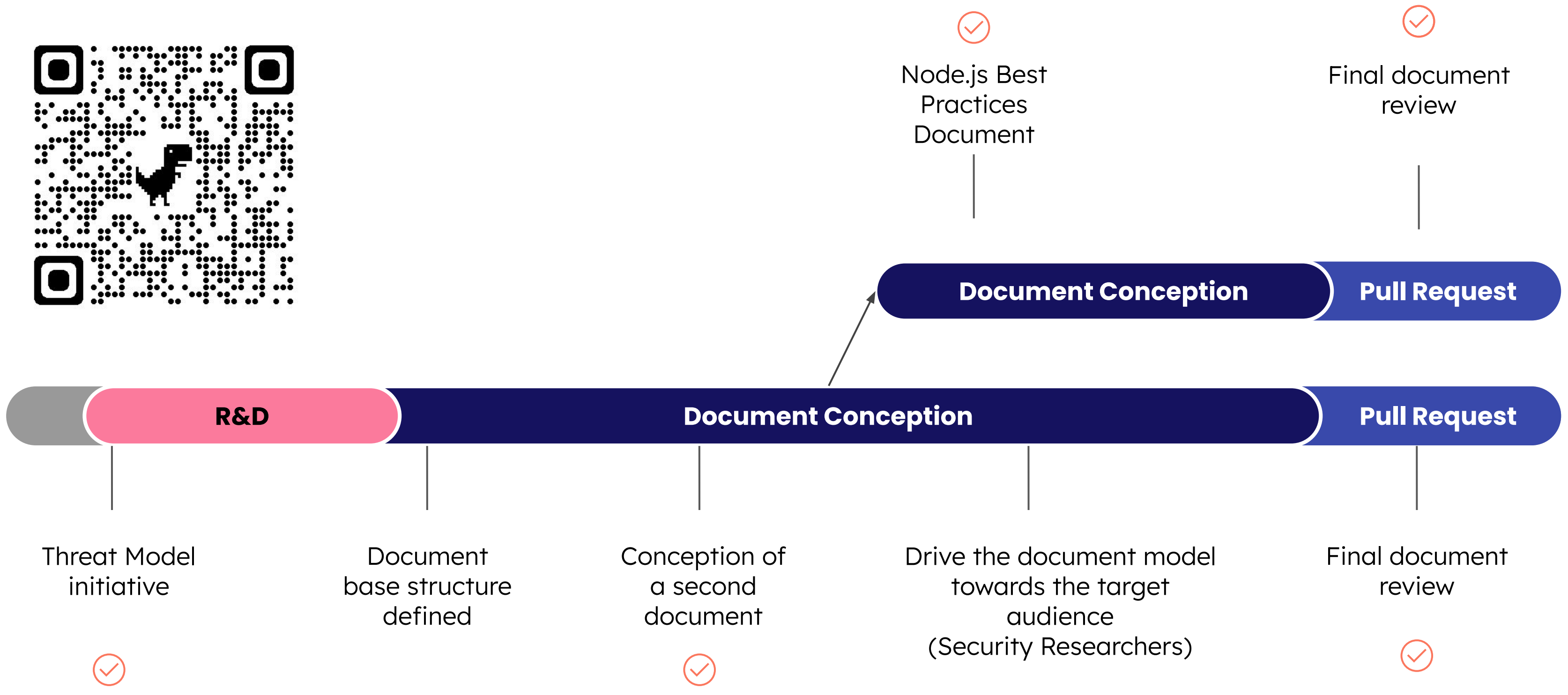


```
process.permission.has('fs.write'); // true
process.permission.has('fs.write', '/home/paulapaul/protected-folder'); // true

process.permission.has('fs.read'); // true
process.permission.has('fs.read', '/home/paulapaul/protected-folder'); // false
```

Security Working Group - Recent Successes

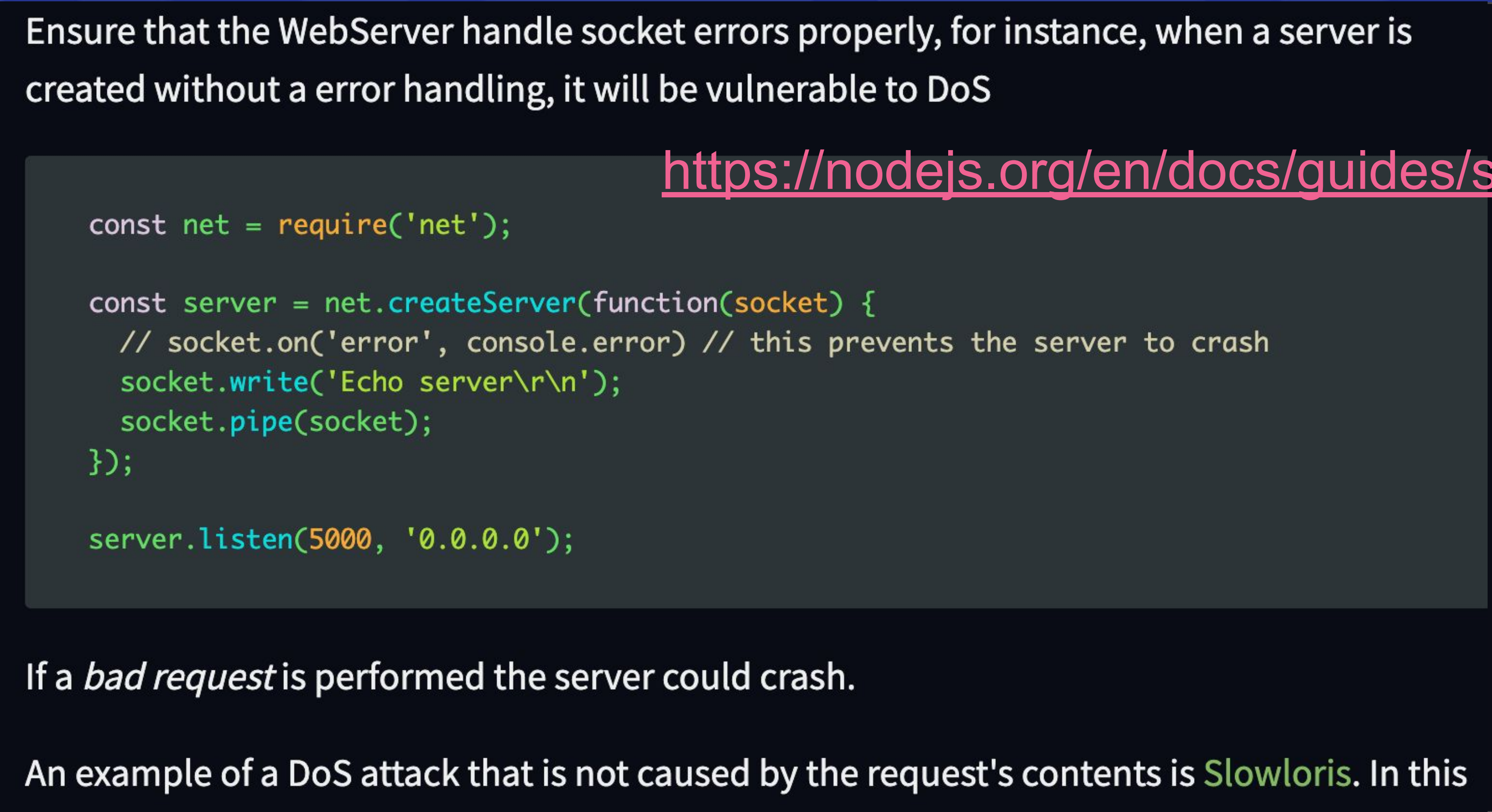
- Threat Model (covered previously)
- Dependency Vulnerability Checks
- Permissions Model
- **Security Best Practices**

Being Proactive: Best Practices - Process & Milestones



✓
Node.js Best Practices Document
✓
Final document review
Document Conception
Pull Request
R&D
Document Conception
Pull Request
Threat Model initiative
✓
Document base structure defined
Conception of a second document
✓
Drive the document model towards the target audience (Security Researchers)
Final document review
✓

Best Practices - Mitigate Denial of Service



Ensure that the WebServer handle socket errors properly, for instance, when a server is created without a error handling, it will be vulnerable to DoS

https://nodejs.org/en/docs/guides/security

```
const net = require('net');

const server = net.createServer(function(socket) {
  // socket.on('error', console.error) // this prevents the server to crash
  socket.write('Echo server\r\n');
  socket.pipe(socket);
});

server.listen(5000, '0.0.0.0');
```

If a *bad request* is performed the server could crash.

An example of a DoS attack that is not caused by the request's contents is Slowloris. In this

Best Practices - Mitigate Prototype Pollution

Prototype pollution refers to the possibility to modify or inject properties into Javascript language items by abusing the usage of *_proto_*, *constructor*, *prototype*, and other properties inherited from built-in prototypes.

```
const a = {"a": 1, "b": 2};
const data = JSON.parse('{"__proto__": { "polluted": true}}');

const c = Object.assign({}, a, data);
console.log(c.polluted); // true

// Potential DoS
const data2 = JSON.parse('{"__proto__": null}');
const d = Object.assign(a, data2);
d.hasOwnProperty('b'); // Uncaught TypeError: d.hasOwnProperty is not a function
```

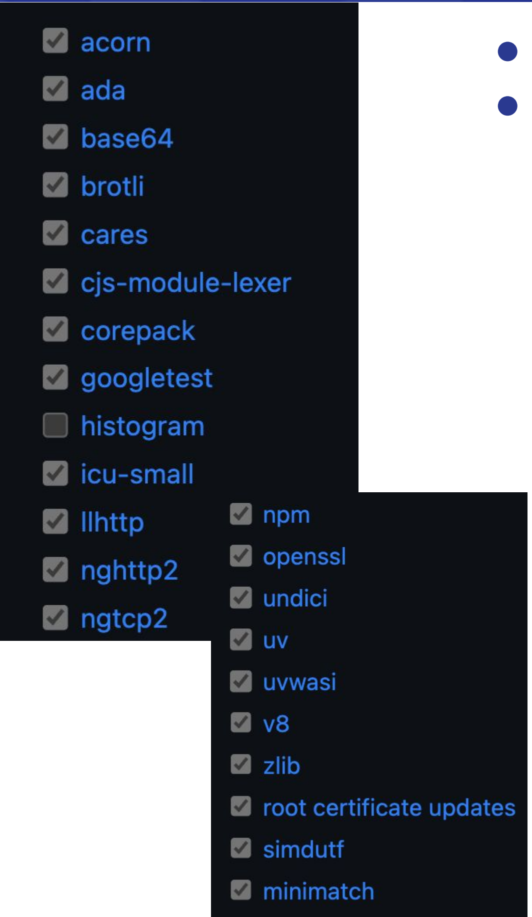
This is a potential vulnerability inherited from the JavaScript language.

Being Proactive: Security WG Ongoing Initiatives

- Automation: dependency updates
- OSSF Scorecard
- Automation: security release process
- Extending the Permission model
- Looking at SigStore and SLSA (just starting)

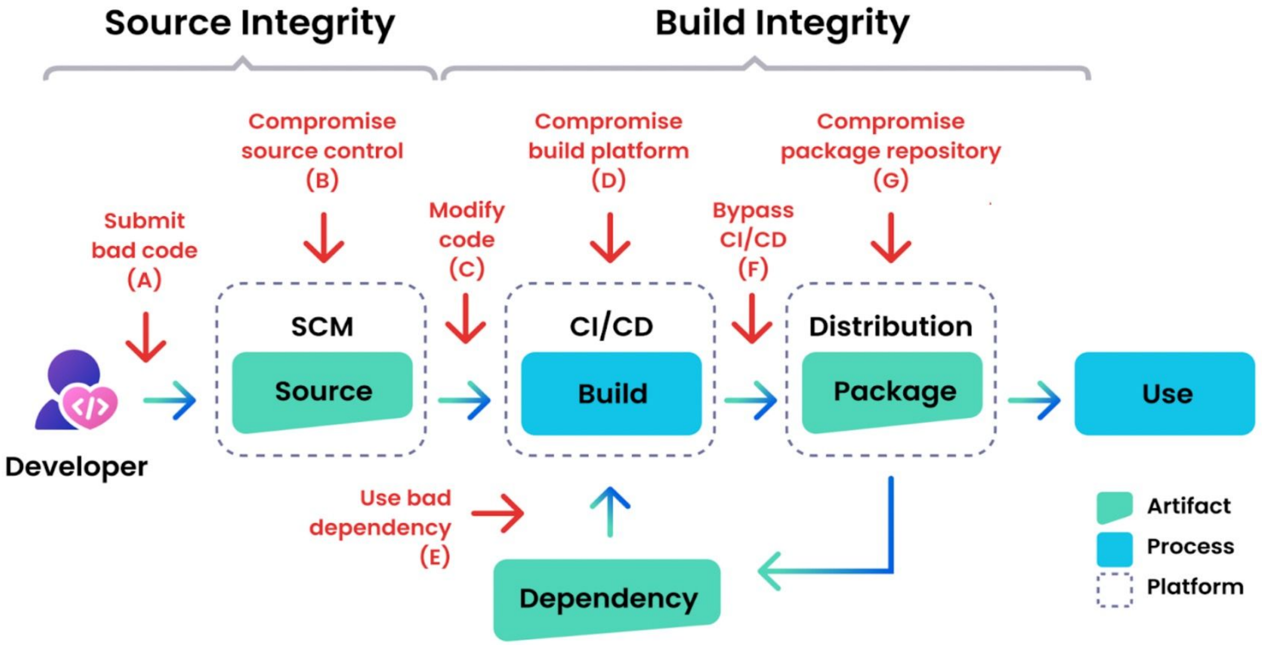
Initiatives:
https://github.com/nodejs/security-wg#current-initiatives

Being Proactive: Automated dependency updates



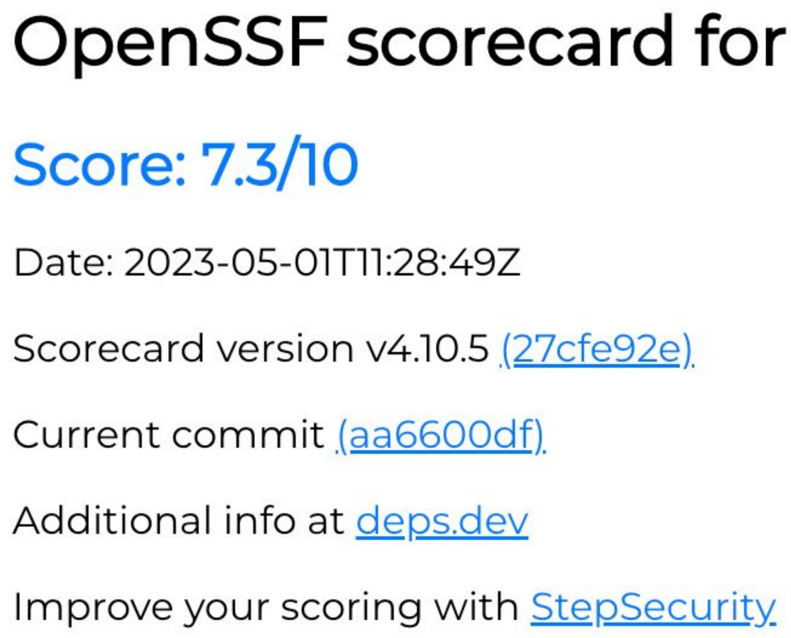
- ☑ acorn
- ☑ ada
- ☑ base64
- ☑ brotli
- ☑ cares
- ☑ cjs-module-lexer
- ☑ corepack
- ☑ googletest
- ☐ histogram
- ☑ icu-small
- ☑ llhttp
- ☑ nghttp2
- ☑ ngtcp2
- ☑ npm
- ☑ openssl
- ☑ undici
- ☑ uv
- ☑ uvwasi
- ☑ v8
- ☑ zlib
- ☑ root certificate updates
- ☑ simdutf
- ☑ minimatch

- Step 1 of improving supply chain security
- **Next**: harden build dependency build and related dependencies



From: https://snyk.io/blog/npm-security-preventing-supply-chain-attacks/

Being Proactive: OSSF Scorecard



OpenSSF scorecard for nodejs/node

Score: 7.3/10

Date: 2023-05-01T11:28:49Z

Scorecard version v4.10.5 (27cfe92e)

Current commit (aa6600df)

Additional info at deps.dev

Improve your scoring with StepSecurity

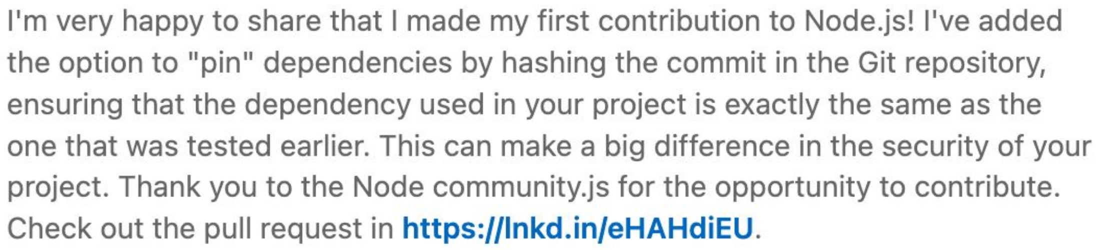
Detailed report with scores and trends by repo, from the Security WG: https://github.com/nodejs/security-wg/blob/main/tools/ossf_scorecard/report.md

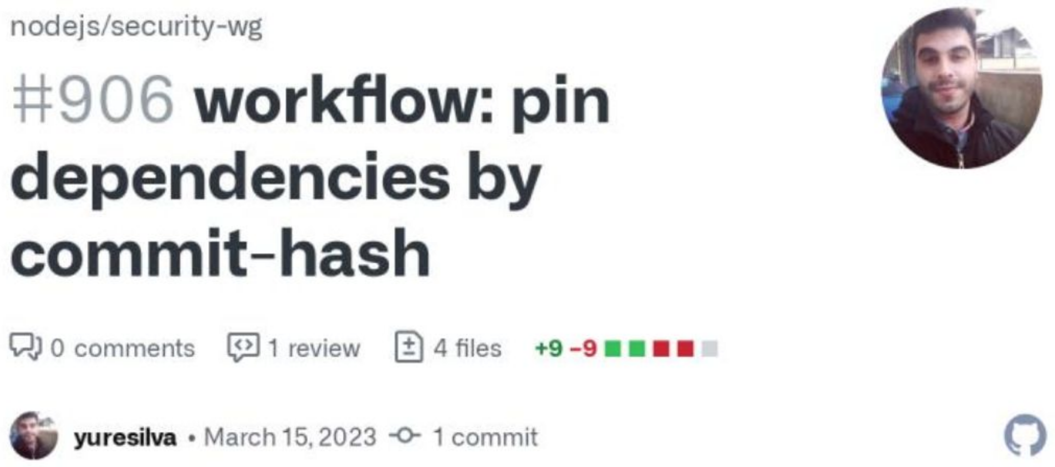
From: https://kooltheba.github.io/openssf-scorecard-api-visualizer/#/projects/github.com/nodejs/node

Being Proactive: OSSF Scorecard

Improving the OSSF Scorecard is a great way to grow security contributors!

Good first issues!


I'm very happy to share that I made my first contribution to Node.js! I've added the option to "pin" dependencies by hashing the commit in the Git repository, ensuring that the dependency used in your project is exactly the same as the one that was tested earlier. This can make a big difference in the security of your project. Thank you to the Node community.js for the opportunity to contribute. Check out the pull request in https://lnkd.in/eHAHdiEU.


nodejs/security-wg

#906 workflow: pin dependencies by commit-hash

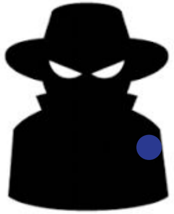
0 comments 1 review 4 files +9 −9 🟩🟩🟥🟥⬜

yuresilva • March 15, 2023 1 commit

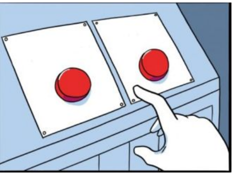
From: https://github.com/nodejs/security-wg/issues/884

Automating security release process

- 26 steps in performing a security release
  - 1 Security Releaser for each Release line
  - 1 Release Steward
  - ~700 hours, ~1 week elapsed time



- Malicious actors don't wait…
  - ***automate*** to improve MTTR!



Normal Release / Security Release

How you can help: Individuals & Organizations

It takes a balance of both!

From: https://veterinaryleadershipinstitute.org/balance-is-key/

How Individuals Can Help: Top six

1. **Contribute** and become a Node.js collaborator
2. **Volunteer** as a security release steward, security triage, or security releaser
3. **Champion** a security working group initiative
4. **Join** the **Security Working Group**
5. **Volunteer** as a security subject matter expert
6. **Contribute** to Security Issues (take on a ‘good first issue’)

Join us at GHC Open Source Day!

Come to a Meeting!

How Organizations Can Help: Top five

1. **Reward people** for helping with triage, fixing vulnerabilities, stewarding and doing security releases
2. **Reward people** for being a security point of contact for your strategic open source dependencies
3. **Implement** vulnerability reporting policies with considerations for open source projects
4. **Join** a foundation that supports Node.js (OpenJS/OpenSSF)
5. **Contribute** to Node.js LFX Bug Bounty/Security Fund

Make a donation! →

Questions?

Copyright and Trademarks