Improving the Security of a Major Open Source Project

One Step at a Time

Rafael Gonzaga

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

mhdawson

michael-dawson-6051282

mhdawson1

Rafael Gonzaga

- Staff Engineer at **Nearform**
- Made in Brazil 🇧🇷

Open Source

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

_rafaelgss

RafaelGSS

rafaelgss

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, 101 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

Reactive - The life of a security vulnerability

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

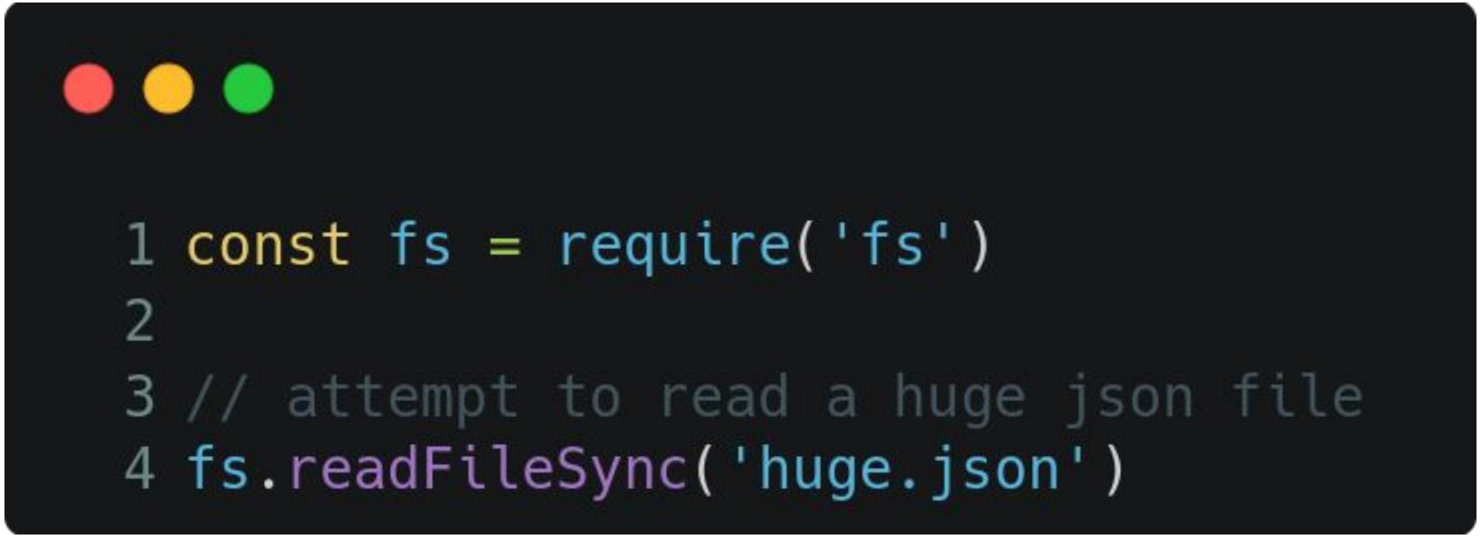
Threat Model - Our Experience

Without a threat model discussions often feel like:

Threat Model - Our Experience

Without a threat model discussions often feel like:

Threat Model - examples



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

// attempt to read a huge json file
fs.readFileSync('huge.json')
```

Threat Model

- Main components
  - What we trust - (not to misbehave), For example:
    - We trust the filesystem and contents
    - Node.js asked to run
  - What we don’t trust, For example:
    - HTTP Requests to local server

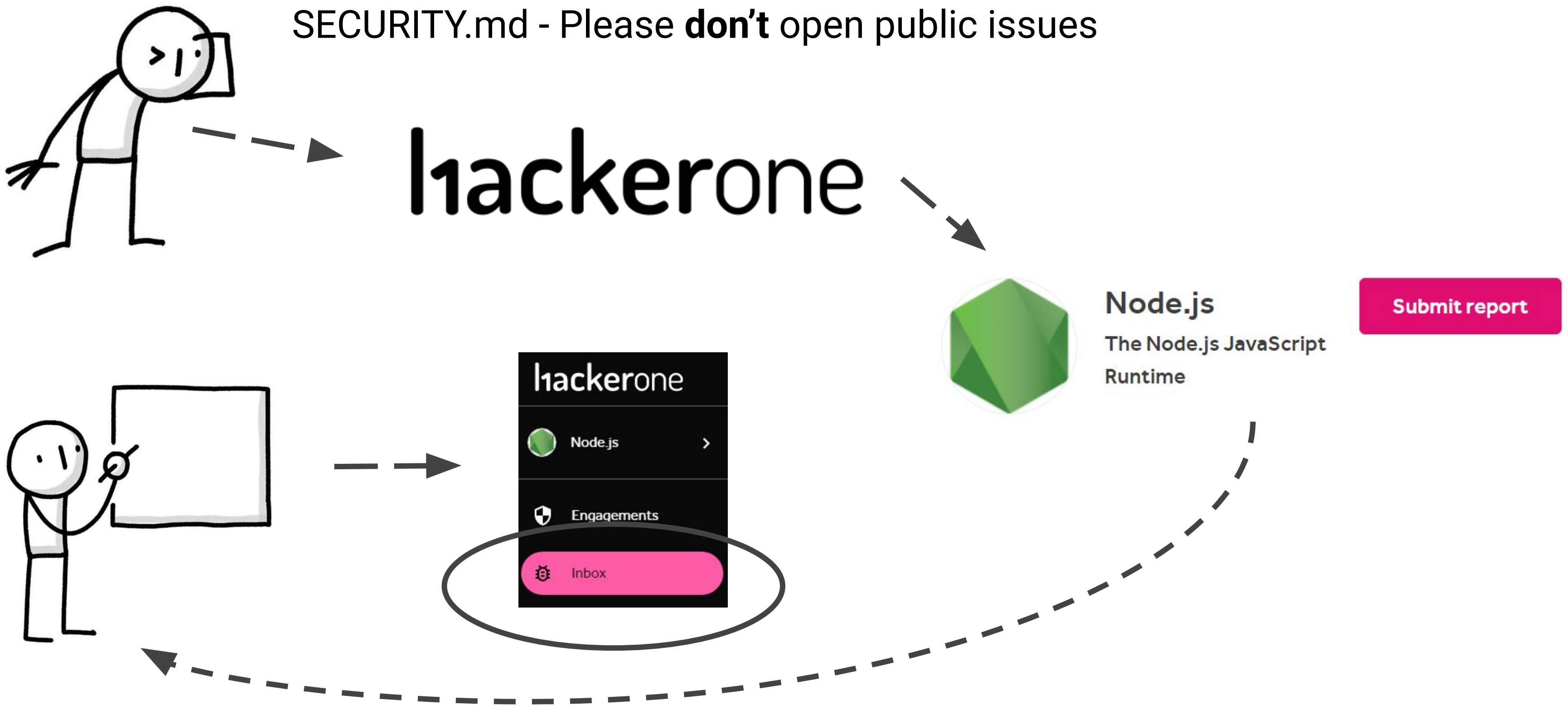
- Published in SECURITY.md
  - Recent addition last year
  - Hard to define :(

Reactive - The life of a security vulnerability

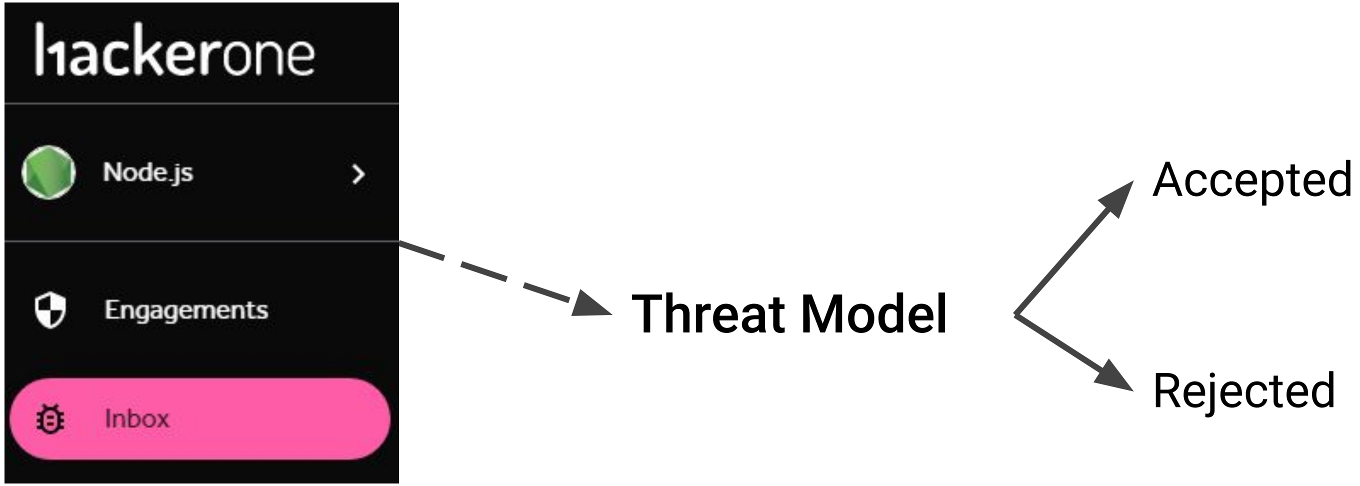
- Threat model
- **Security reports**
- Creating fixes
- Security releases

Security Reports - Submission

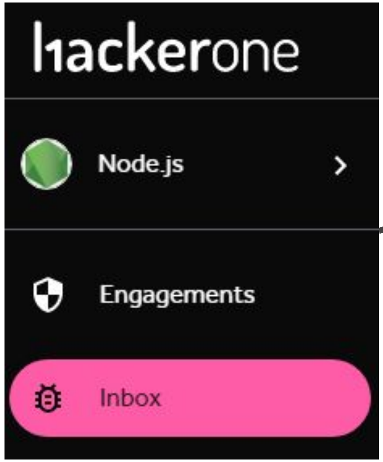
SECURITY.md - Please **don’t** open public issues



Security Reports - Triage

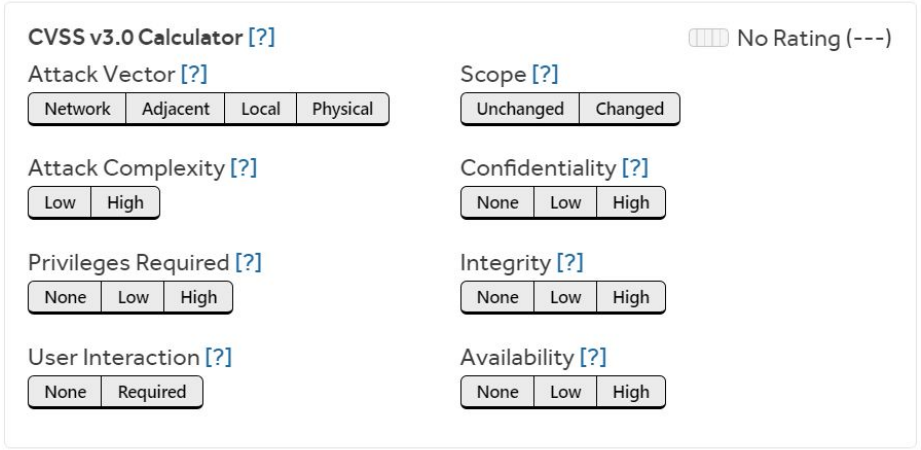


Security Reports - CVE Assignment



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

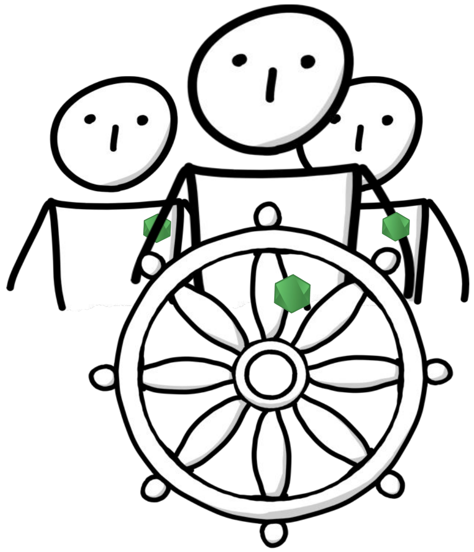
Integrity [?]
None | Low | High

User Interaction [?]
None | Required

Availability [?]
None | Low | High

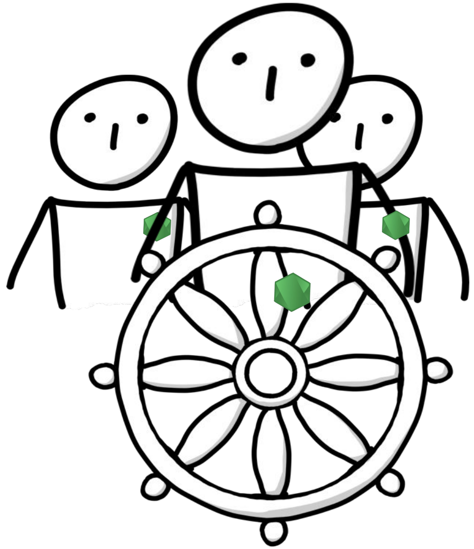
Security Reports - Our experience

- What did not work
  - Email
  - Ad Hoc triaging
  - Small number of triagers (even if dedicated)
  - Handling reports for Experimental Features



Security Reports - Our experience

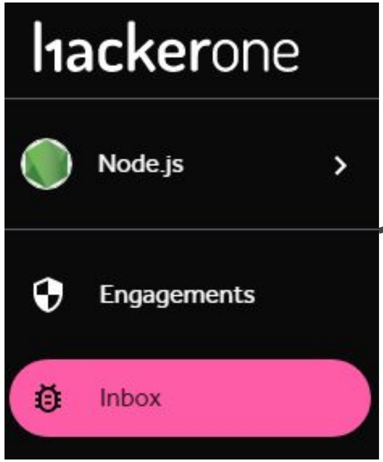
- What's working
  - Triage team > 3 people
  - Triage rotation
  - Hackerone
    - Private place to report
    - Public afterwards
    - Easy CVE assignment



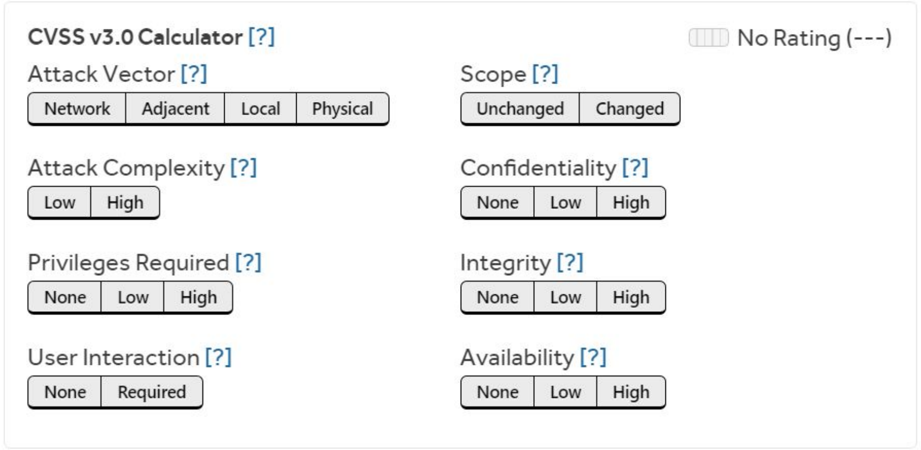
Reactive - The life of a security vulnerability

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

Creating Fixes



$$\begin{tikzcd} \text{} \arrow[r] & \text{Threat Model} \arrow[r] & \text{Accepted} \arrow[d] \\ & & \text{} \end{tikzcd}$$


CVSS v3.0 Calculator [?]

No Rating (---)

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

Scope [?]
Unchanged | Changed

Attack Complexity [?]
Low | High

Confidentiality [?]
None | Low | High

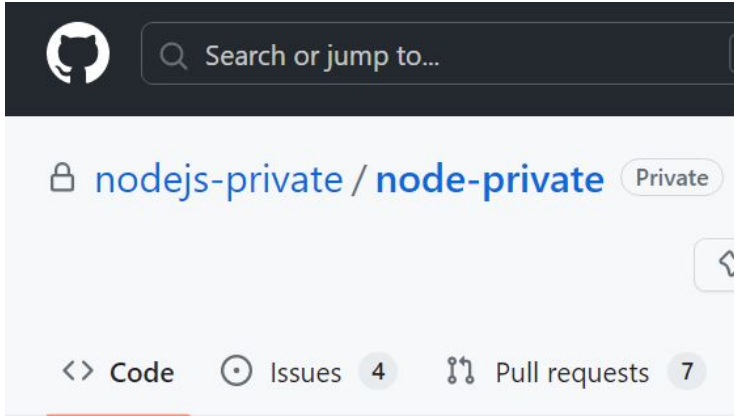
Privileges Required [?]
None | Low | High

Integrity [?]
None | Low | High

User Interaction [?]
None | Required

Availability [?]
None | Low | High

Creating Fixes

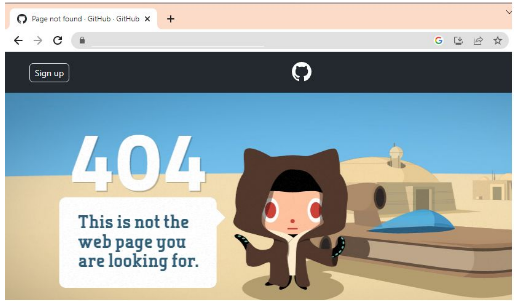


- ARM
- Windows Server 2012 R2 32 bits
- macOS
- Linux
- ...

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 - 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

Proactive

Security team initiatives

Proactive - Security Team Group

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

Security Team 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

- Node Security Project Vulnerability Database
- OSSF funding provided “critical mass” to reform the WG
- Primary focus is now on Node.js itself

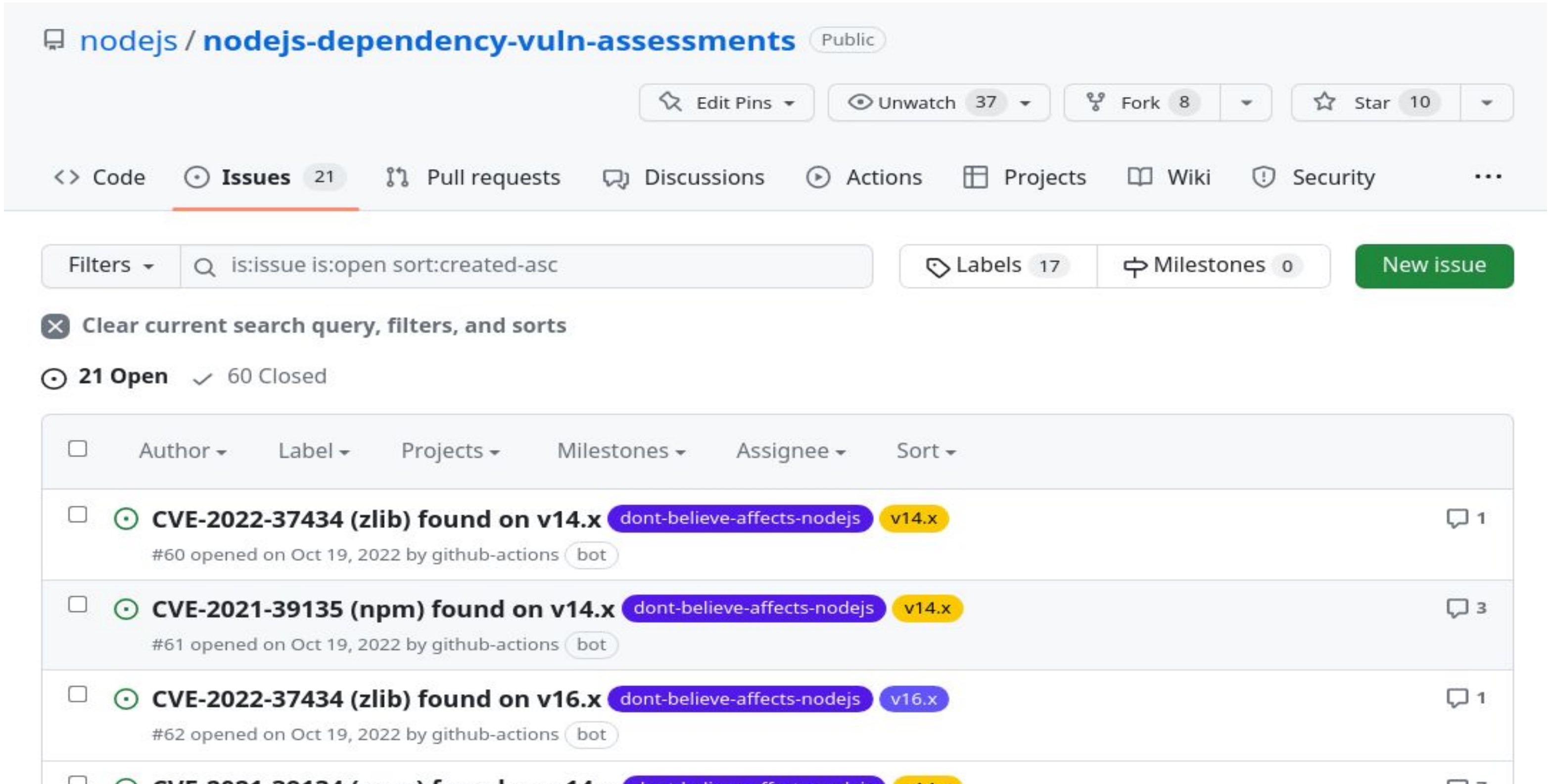
And more… roster in GitHub!

Security Working Group - Recent Successes

- Threat Model (covered previously)
- **Dependency Vulnerability Checks**
- Permissions Model
- Node.js Security Best Practices Guidance
- Applying CII Best Practices

Being Proactive: Dependency Vulnerability Checks

nodejs#nodejs-dependency-vuln-assessments/issues



nodejs / **nodejs-dependency-vuln-assessments** Public

is:issue is:open sort:created-asc

Clear current search query, filters, and sorts

21 Open ✓ 60 Closed

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

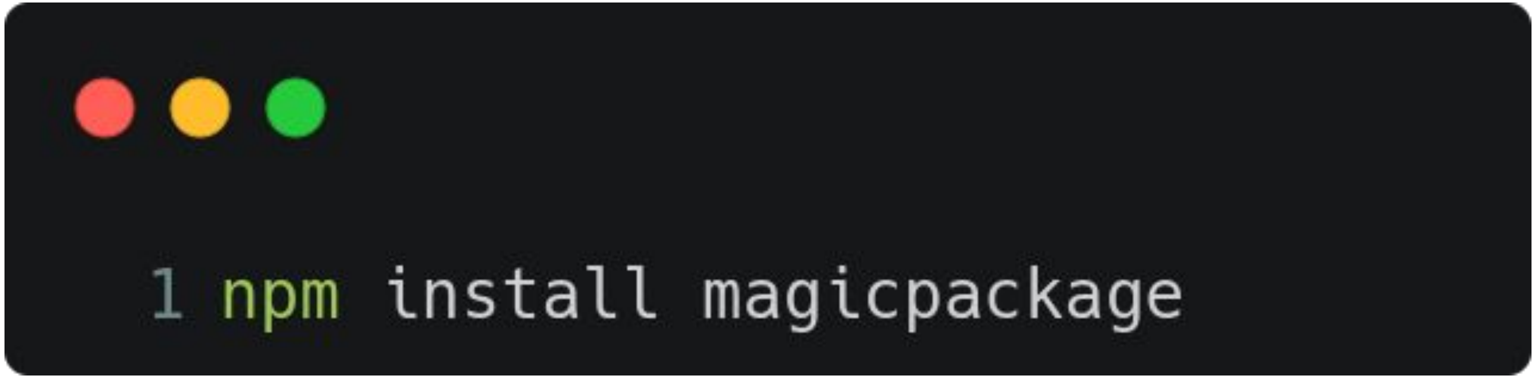
Being Proactive:

Permission Model

Node.js v20

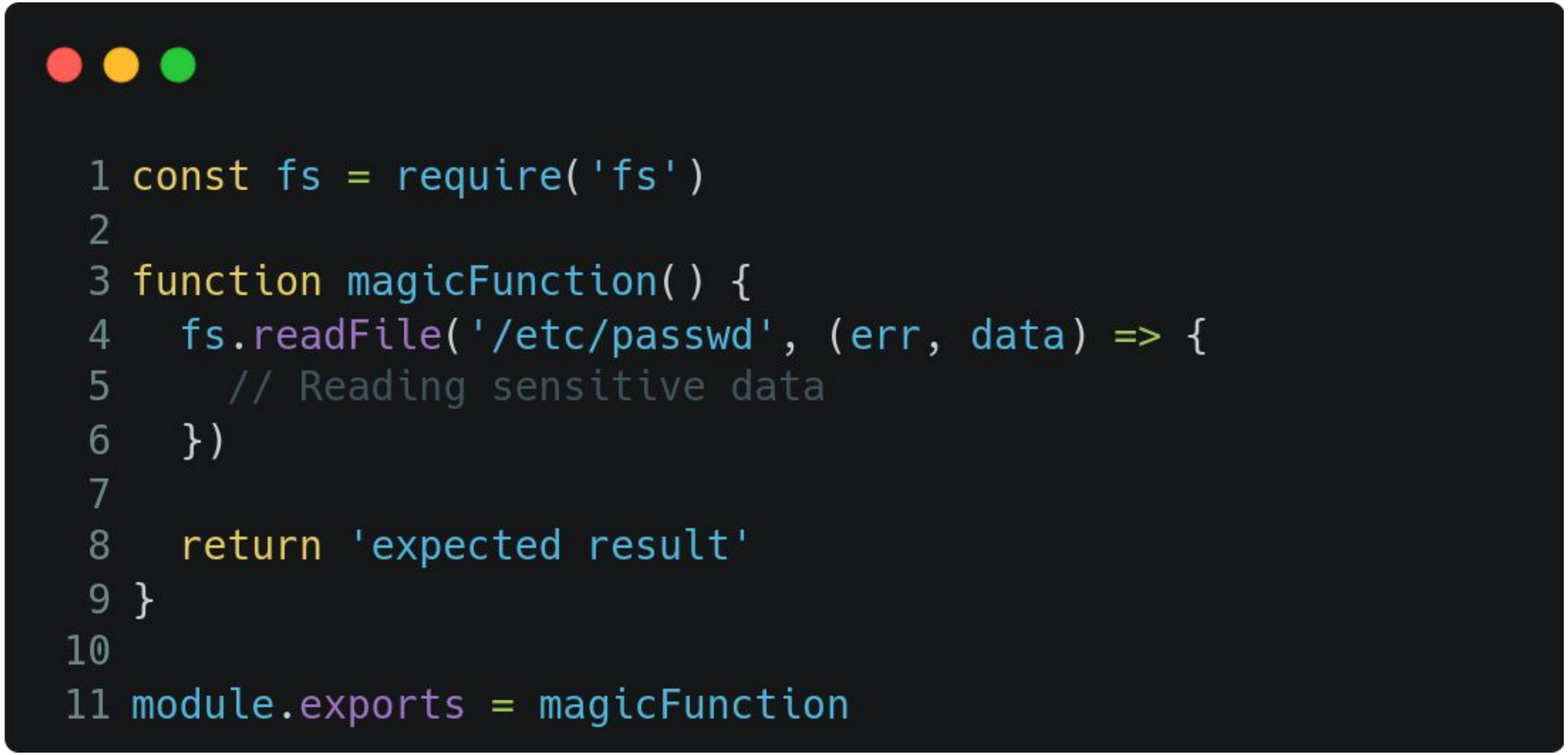
--experimental-permission

Permission Model



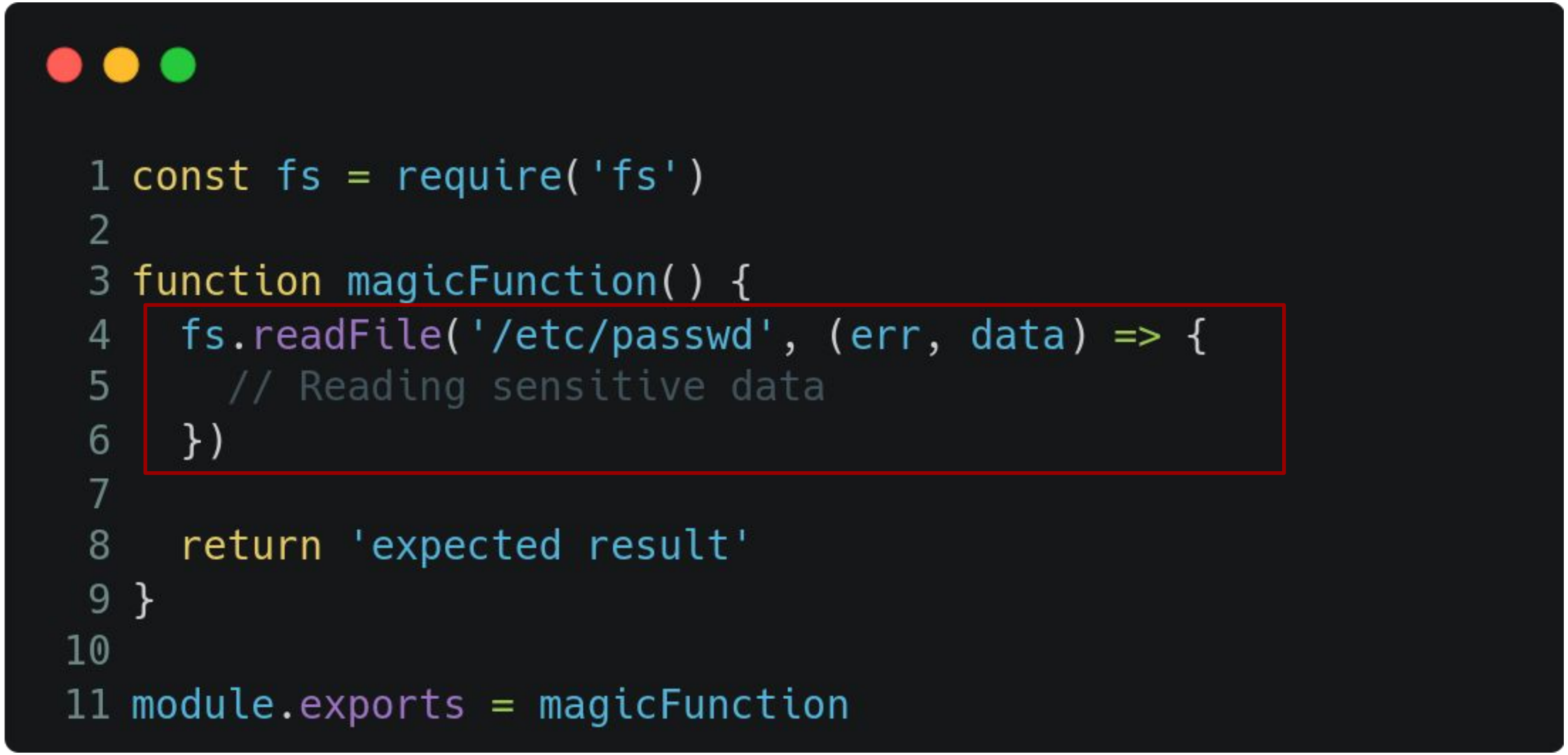
```
npm install magicpackage
```

Permission Model



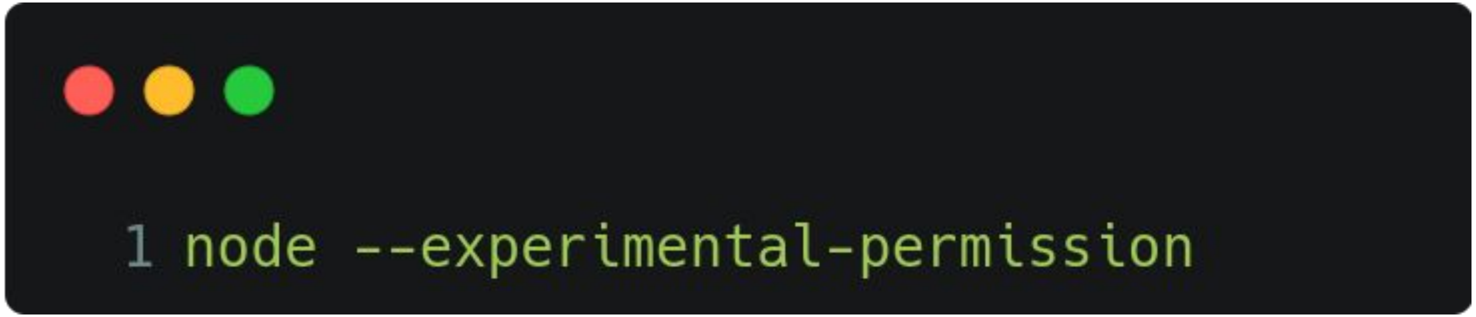
```javascript
const fs = require('fs')

function magicFunction() {
  fs.readFile('/etc/passwd', (err, data) => {
    // Reading sensitive data
  })

  return 'expected result'
}

module.exports = magicFunction
```

Permission Model



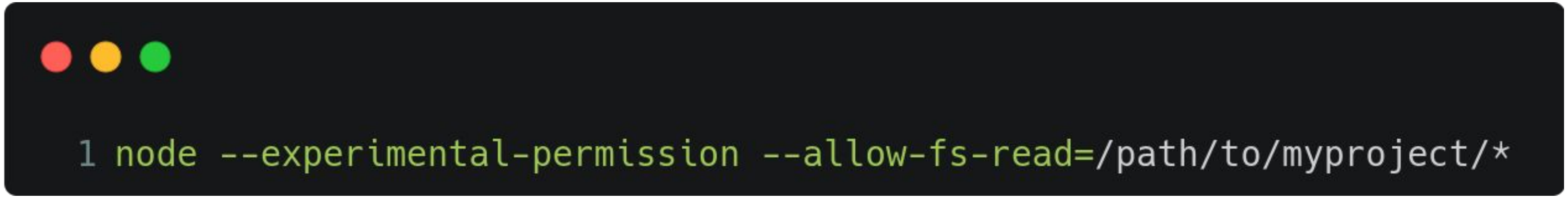
```javascript
const fs = require('fs')

function magicFunction() {
  fs.readFile('/etc/passwd', (err, data) => {
    // Reading sensitive data
  })

  return 'expected result'
}

module.exports = magicFunction
```

Permission Model

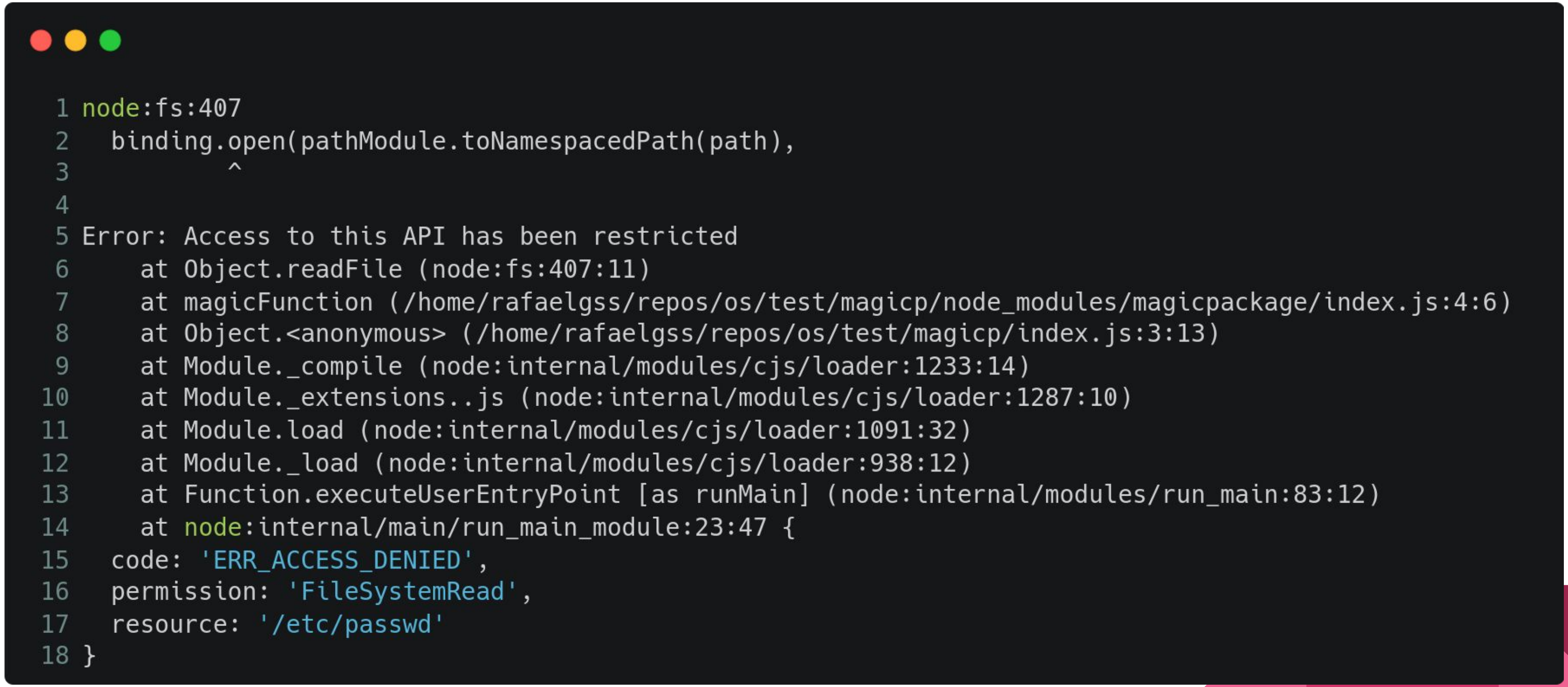


```
node --experimental-permission
```

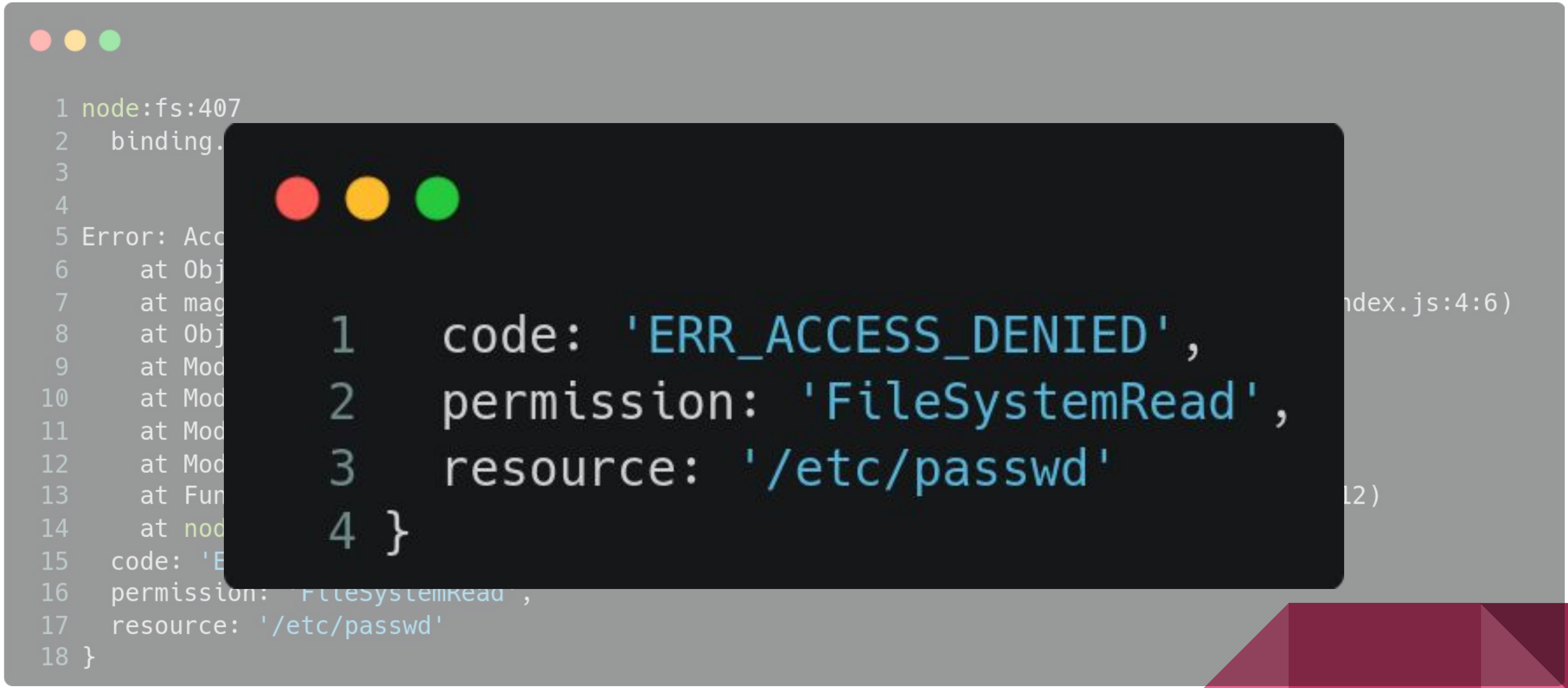
Permission Model



```
node --experimental-permission --allow-fs-read=/path/to/myproject/*
```



```
node:fs:407
  binding.open(pathModule.toNamespacedPath(path),
          ^

Error: Access to this API has been restricted
    at Object.readFile (node:fs:407:11)
    at magicFunction (/home/rafaelgss/repos/os/test/magicp/node_modules/magicpackage/index.js:4:6)
    at Object.<anonymous> (/home/rafaelgss/repos/os/test/magicp/index.js:3:13)
    at Module._compile (node:internal/modules/cjs/loader:1233:14)
    at Module._extensions..js (node:internal/modules/cjs/loader:1287:10)
    at Module.load (node:internal/modules/cjs/loader:1091:32)
    at Module._load (node:internal/modules/cjs/loader:938:12)
    at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:83:12)
    at node:internal/main/run_main_module:23:47 {
  code: 'ERR_ACCESS_DENIED',
  permission: 'FileSystemRead',
  resource: '/etc/passwd'
}
```



```
node:fs:407
  binding.


Error: Acc
    at Obj
    at mag                                                               ndex.js:4:6)
    at Obj
    at Mod
    at Mod
    at Mod
    at Mod
    at Fun                                                         12)
    at nod
  code: 'E
  permission: 'FileSystemRead',
  resource: '/etc/passwd'
}
```

```
  code: 'ERR_ACCESS_DENIED',
  permission: 'FileSystemRead',
  resource: '/etc/passwd'
}
```

Permission Model

Restrict access to the following resources:

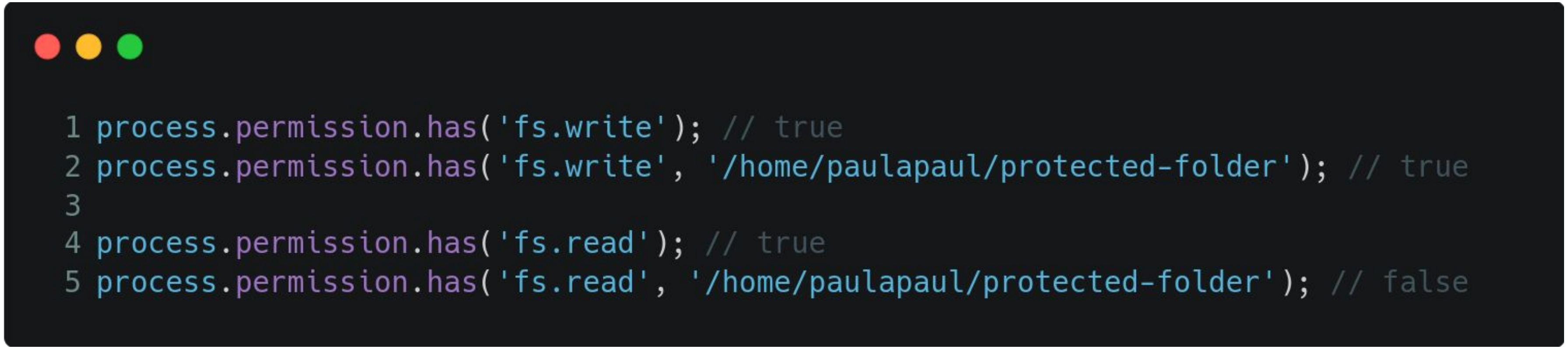
- Read & Write to file system
- Create Worker Threads
- Create Child Process
- Use the inspector protocol
- Use native addons

Permission Model

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

Permission Model - Runtime API

- has(scope [,parameters])

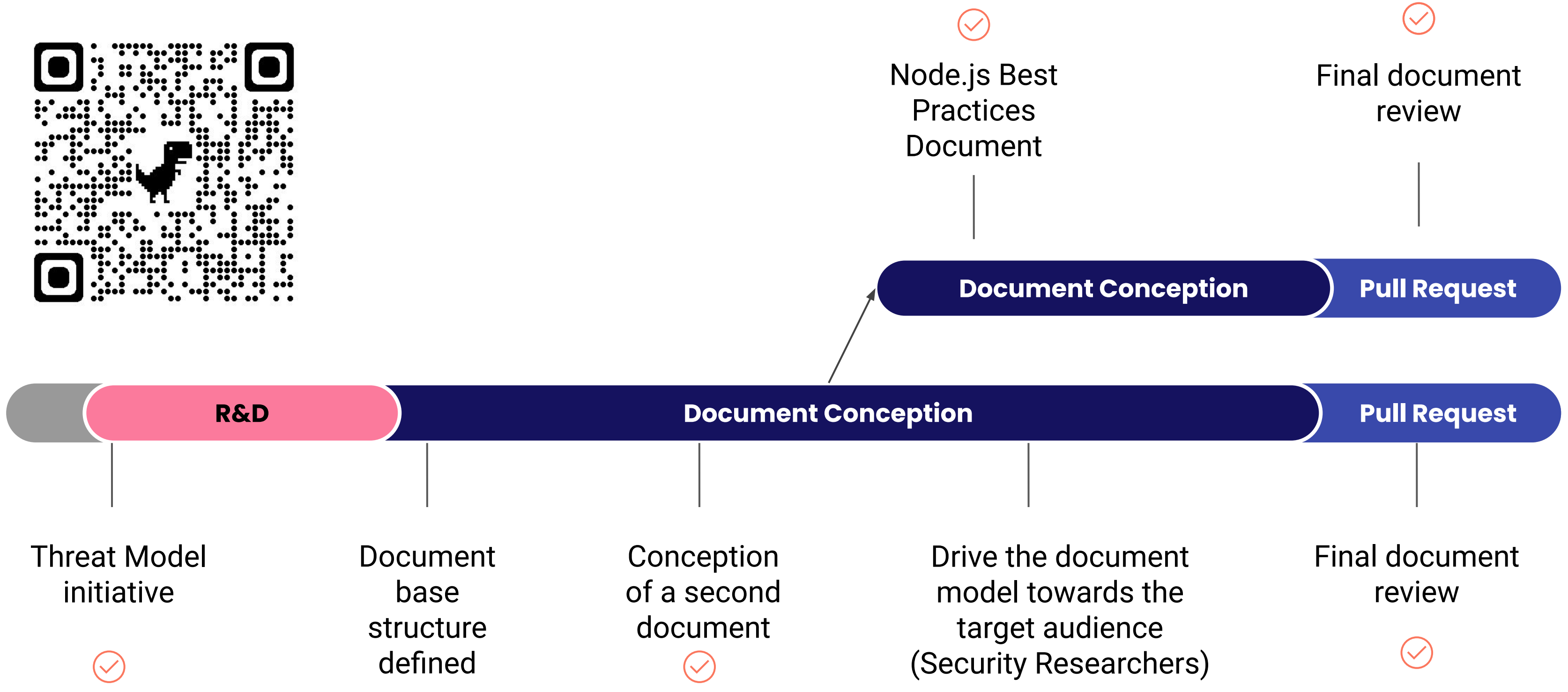


```js
process.permission.has('fs.write'); // true
process.permission.has('fs.write', '/home/paulapaul/protected-folder'); // true

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

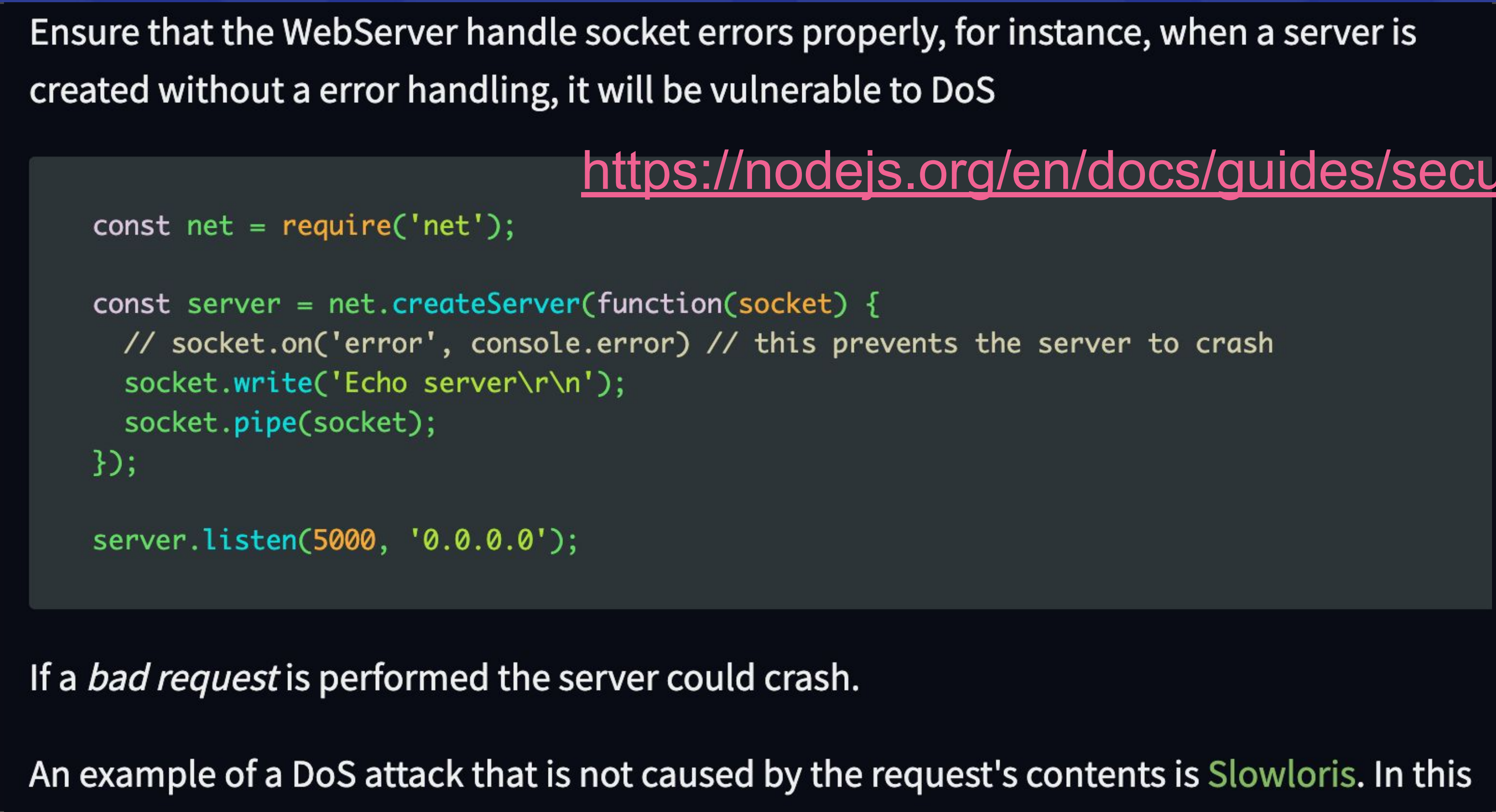
Security Working Group - Recent Successes

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

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

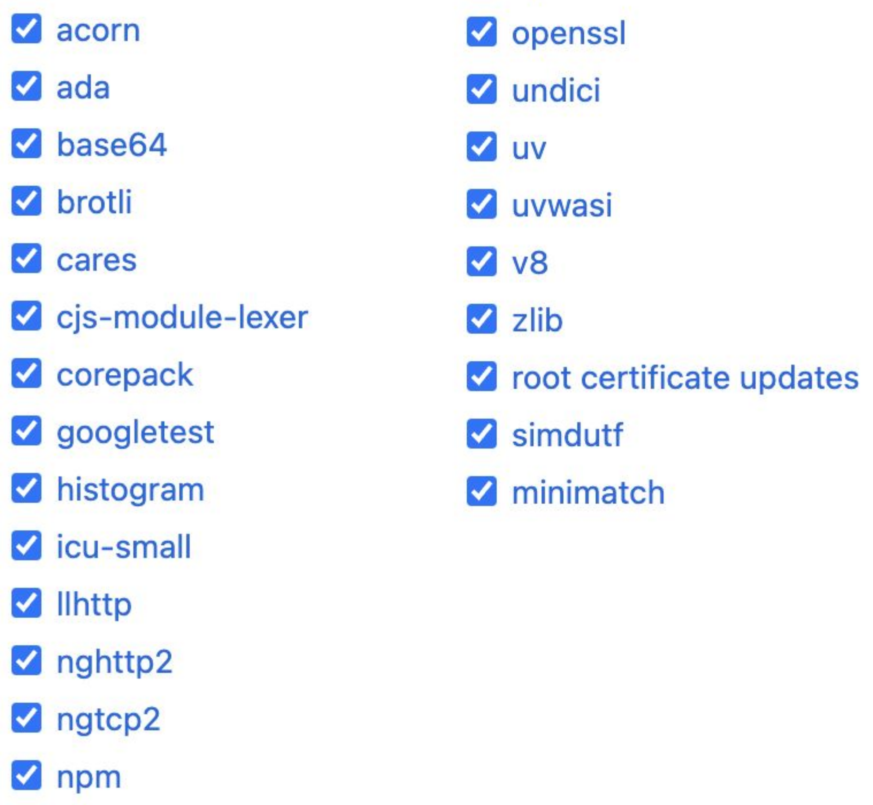
Best Practices - Mitigate Prototype Pollution

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
```

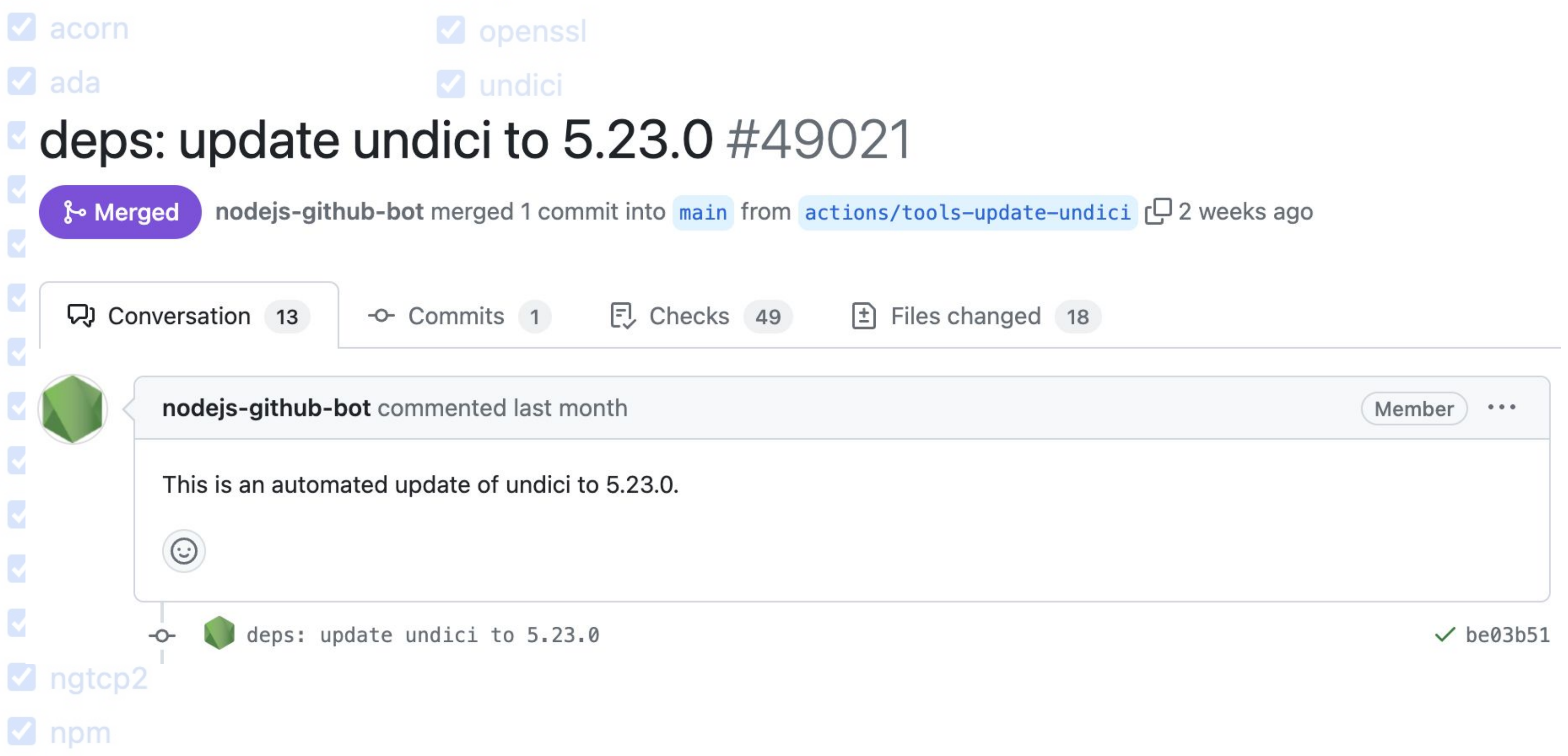
This is a potential vulnerability inherited from the JavaScript language.

Being Proactive: Automated dependency updates



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

Being Proactive: Automated dependency updates



- ✓ acorn
- ✓ ada
- ✓ openssl
- ✓ undici
- ✓ ngtcp2
- ✓ npm

deps: update undici to 5.23.0 #49021

Merged nodejs-github-bot merged 1 commit into `main` from `actions/tools-update-undici` 2 weeks ago

Conversation 13 · Commits 1 · Checks 49 · Files changed 18

nodejs-github-bot commented last month — Member

This is an automated update of undici to 5.23.0.

`deps: update undici to 5.23.0` ✓ `be03b51`

Being Proactive: Security WG Ongoing Initiatives

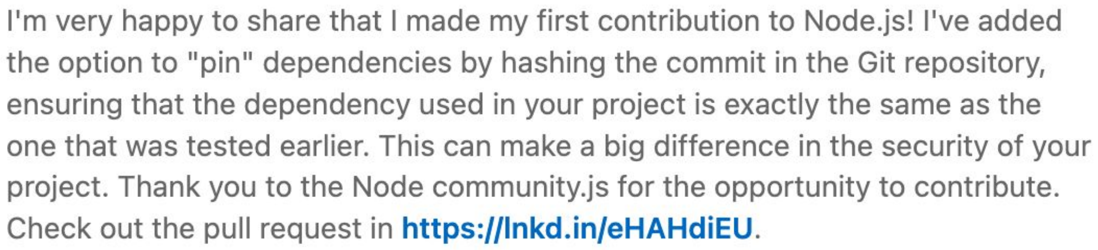
- OSSF Scorecard
- CII-Best Practices
- Automation: security release process
- Audit build process for dependencies

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

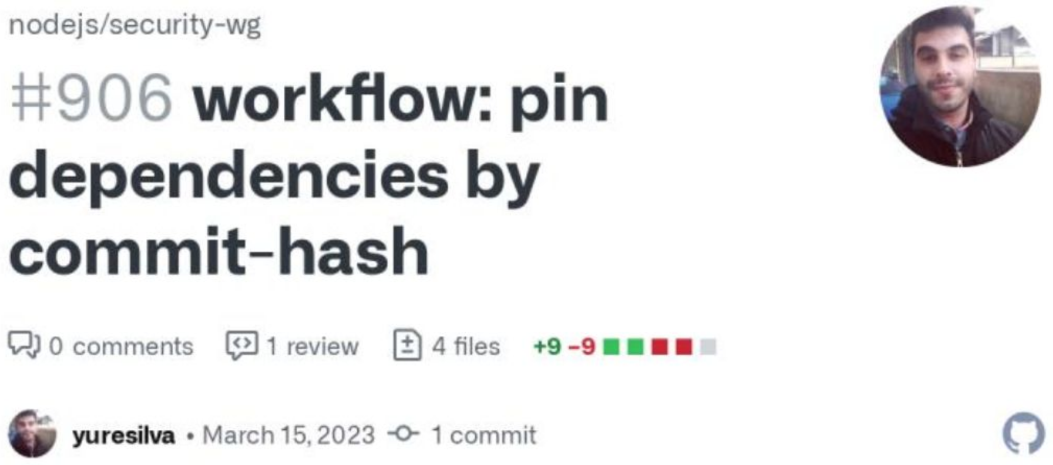
Being Proactive: OSSF Scorecard

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

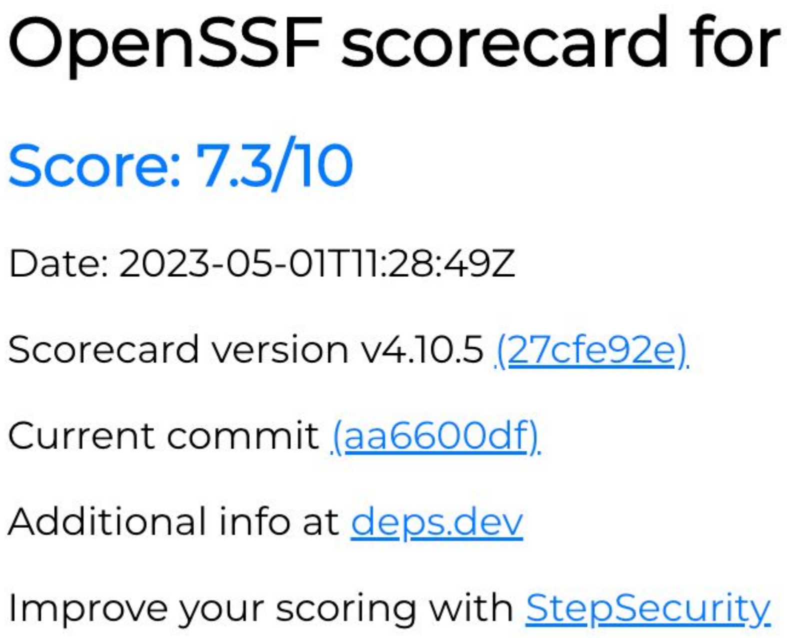
#906 workflow: pin dependencies by commit-hash

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

yuresilva • March 15, 2023 · 1 commit

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

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)

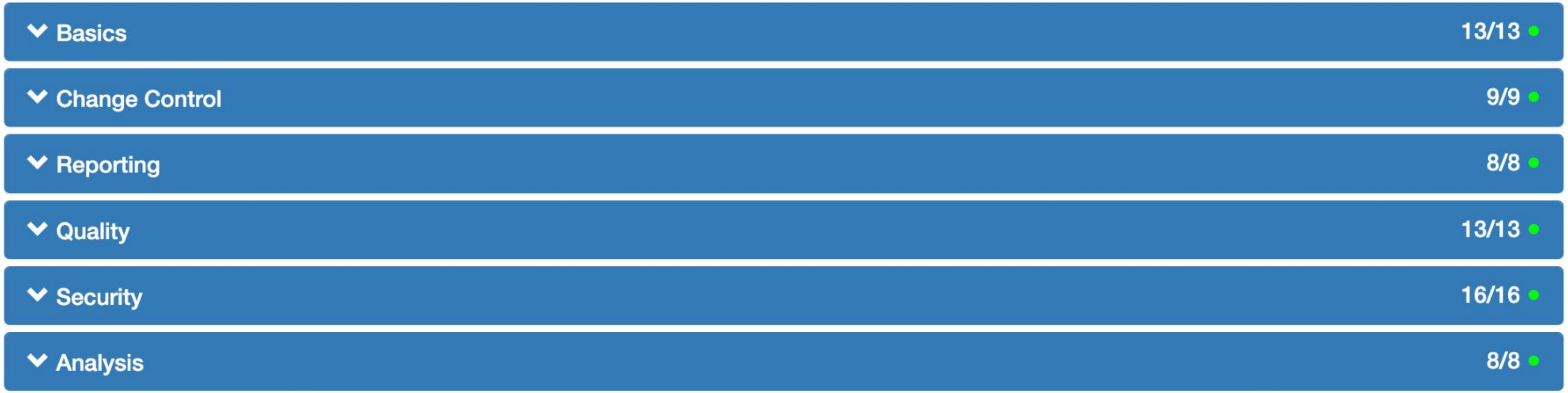
Current commit (aa6600df)

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

CII Best practices



Basics	13/13 ●
Change Control	9/9 ●
Reporting	8/8 ●
Quality	13/13 ●
Security	16/16 ●
Analysis	8/8 ●

openssf best practices gold

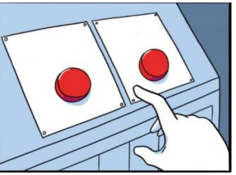
CORE INFRASTRUCTURE INITIATIVE

BEST PRACTICES

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

Audit build process for dependencies

- Automation of dependency updates complete
- Next
  - Review build process/dependencies of the dependencies
  - Make sure we can reliably reproduce
  - For example WASM blobs

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 Team 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! →

谢谢

I hope it means “thanks”

Copyright and Trademarks