



Pull What Where?

Contributing to Open Source, Securely, on GitHub

Jamie Jones

  @jbjonesjr

GitHub Principal Architect & Regulated Industries Solutions Lead

- * GitHub for over three years
- * Previously Developer, Technical Lead & Configuration Manager with Security-conscious Federal agencies.
- * Casual Open Source enthusiast
- * ❤️ The outdoors and landscape photos





FINOS

Open Source Readiness Program

- Regulatory Compliance WG Chair

FinServ Developer Experience Program Chair

- Developer Experience WG Chair

Goals

- Overview of the value of open source
- Understanding the different workflows used publishing code outside your organization
- Understanding GitHub and integration points key for process security
- Better practices for open source security
- Things to think about when staying legal and compliant with open source code
- Critical tools to integrate into your process





“Ten or 15 years ago, the engineers were the ones who didn’t speak to anyone and maybe seemed like they hadn’t showered that day. The traders were the ones that looked like they stepped out of a Brooks Brothers catalog,” said. “That line has basically disappeared.”

– Oliver Cooke, a financial-industry recruiter at Selby Jennings



“Everyone who comes to sales and trading needs to know how to code.”

– Adam Korn, Goldman Sachs

OSS usage continues to grow

What percent of the typical app you work on is comprised of open source components?

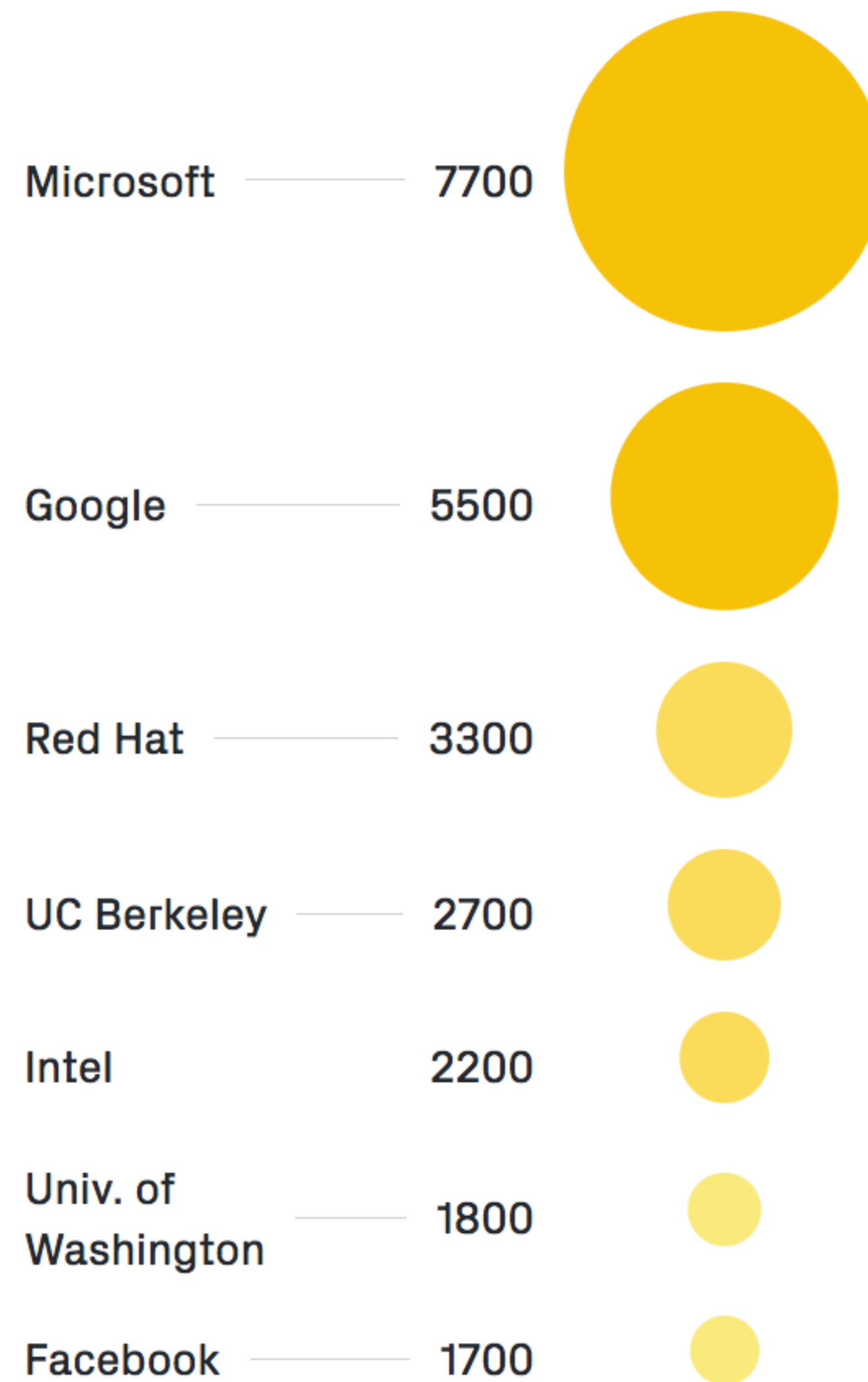
What percentage of Apps include ANY open source?

0-49%	64.4% (13,960,231)
50-74%	27.8% (6,122,356)
75-100%	8.8% (1,935,860)
Total	22,018,447

96%

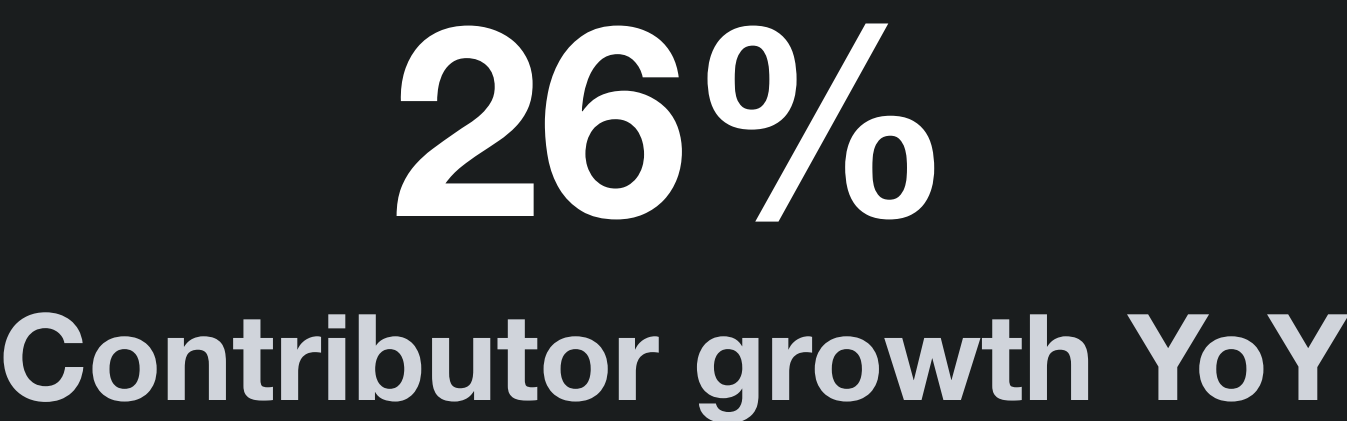
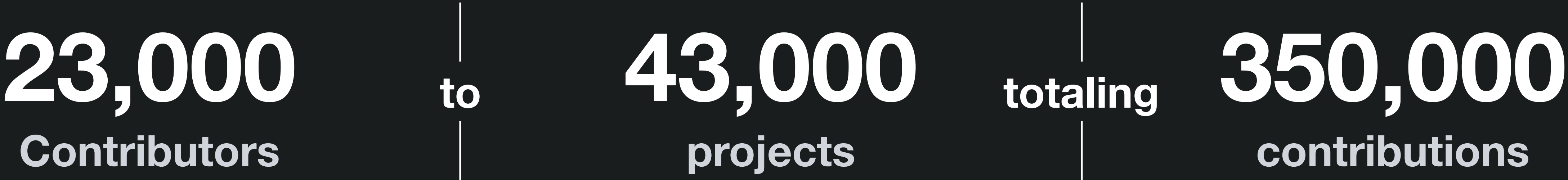
* Evans Data Research Survey - Global Developer Survey 2017





Top open source organizations

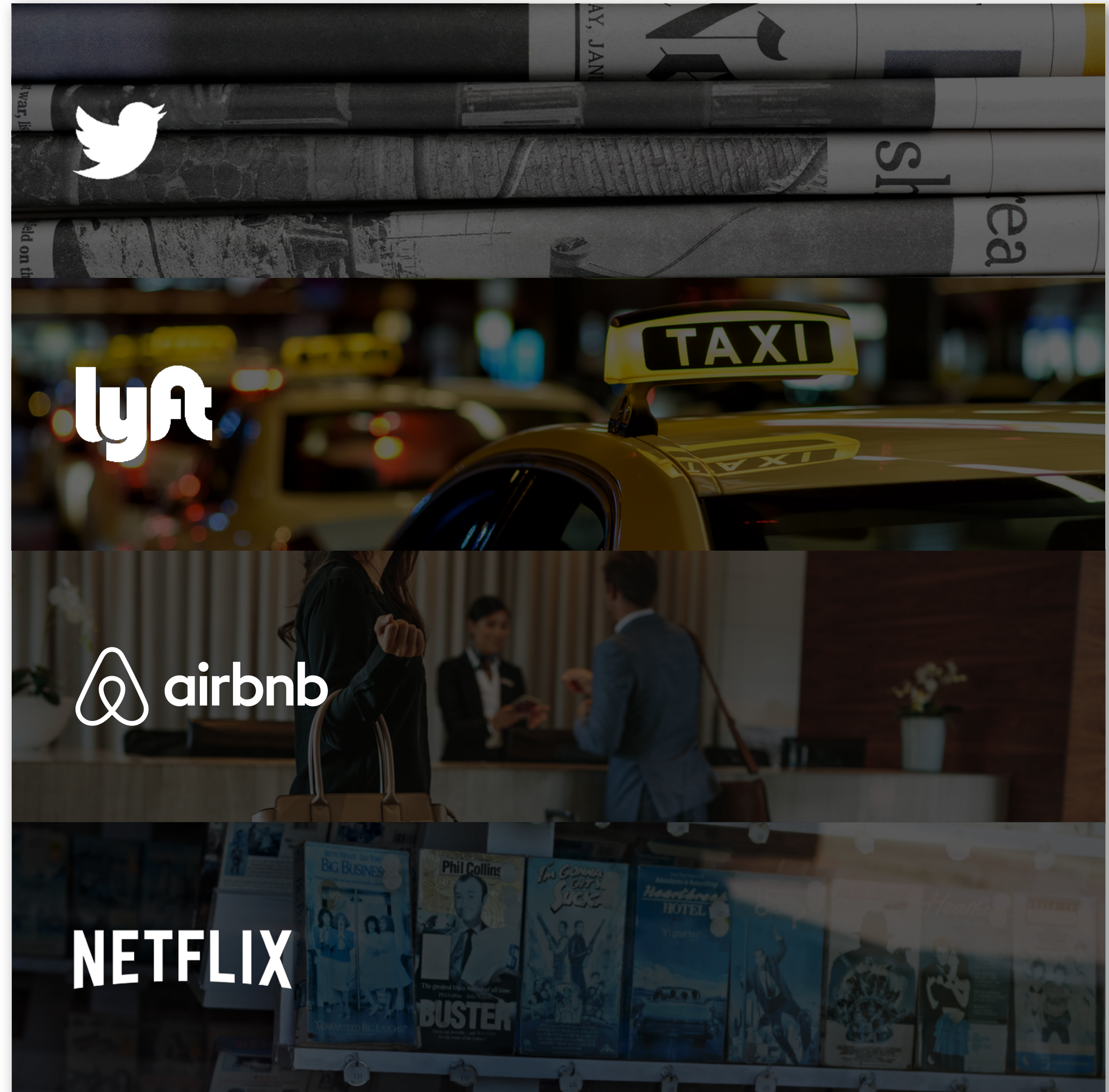
2017 - Open Source in Financial Services



Nearly 50% of the current
S&P 500
will be replaced
by 2026

Innosight

Corporate Longevity: Turbulence Ahead
for Large Organizations



Open Source Workflows

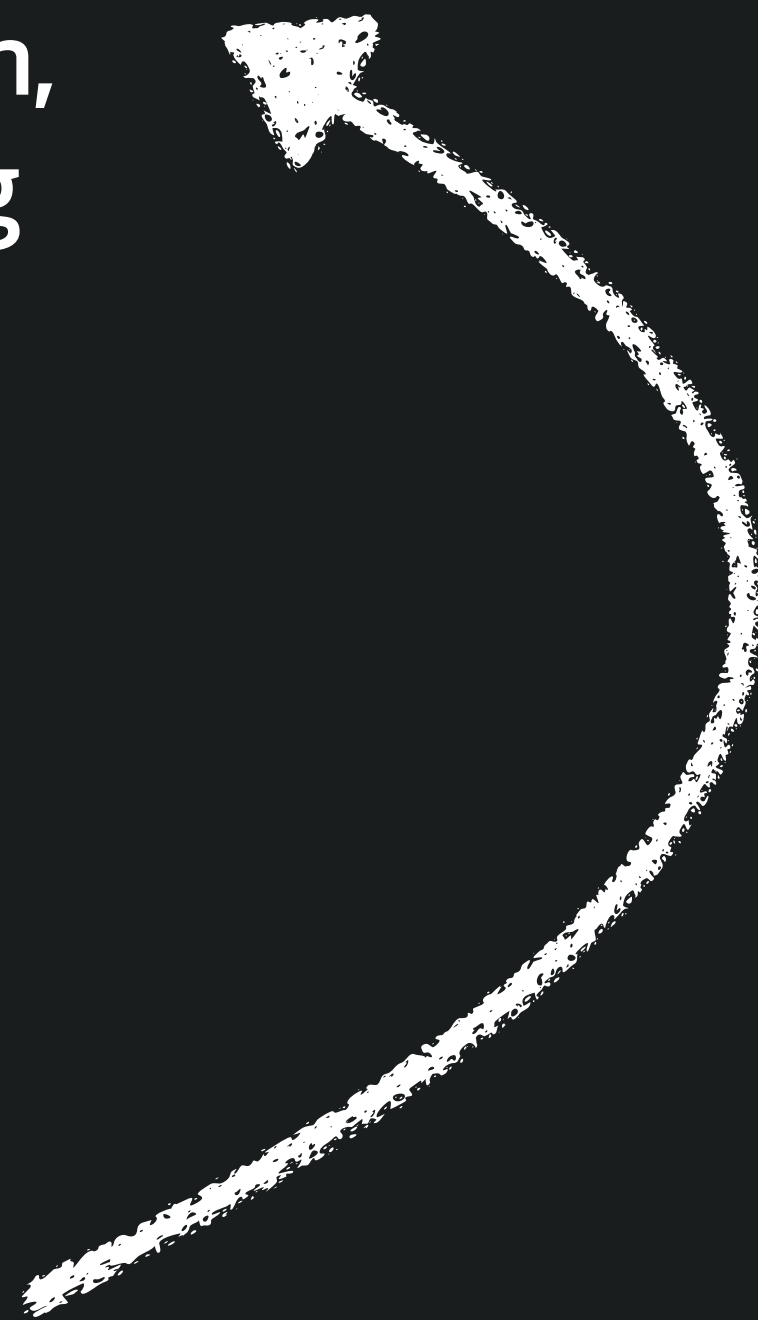


Code in the open!!!!



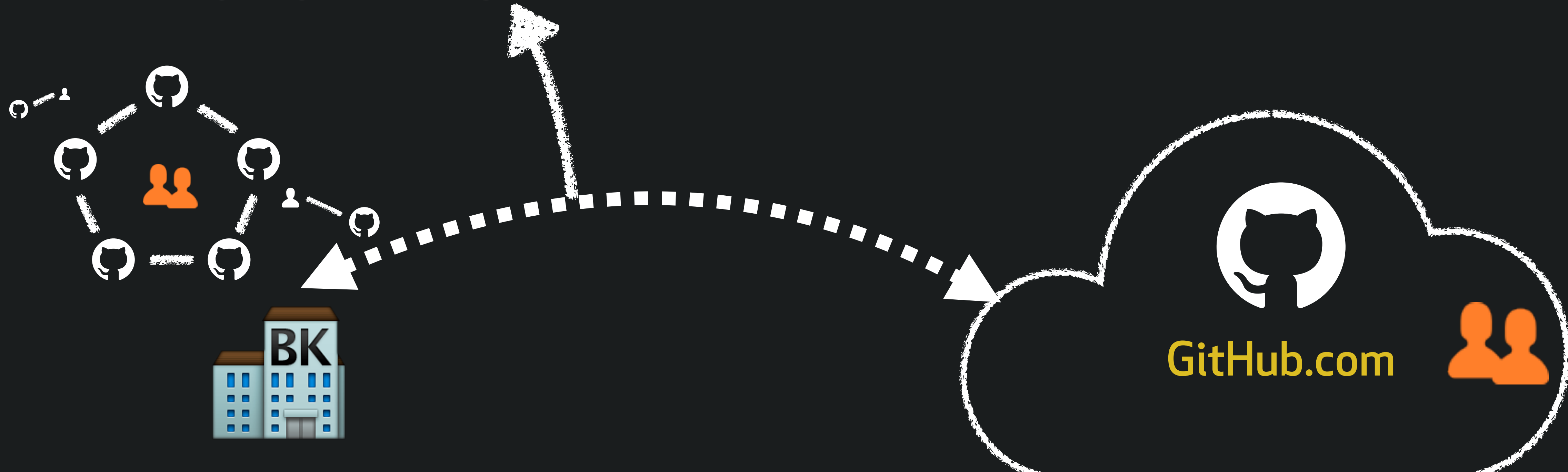
Do it on your own time!

- Process enforcement
- Leadership Approvals
- Security
 - Pii, Dependency Checking, Dirty Words, Token Scanning, Static and Dynamic inspection, ALMSS, Vulnerability tracking
- Compliance & Audit



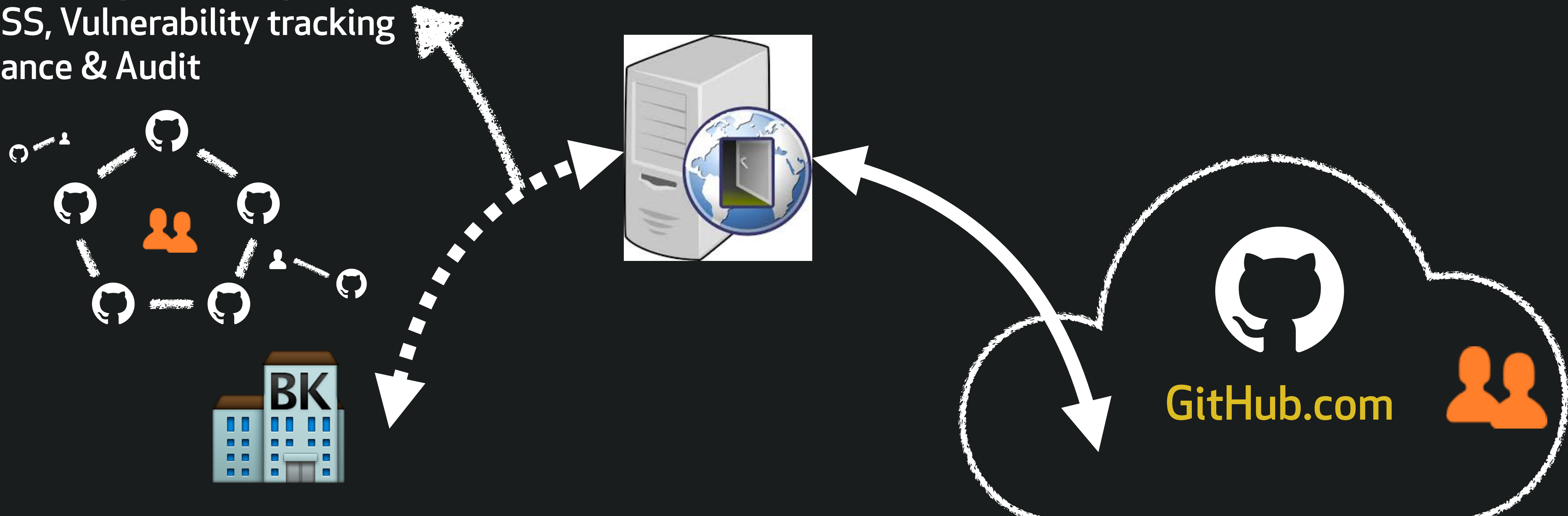
Okay, we'll integrate... with a process

- Filling out forms
- Getting approvals
- Waiting days (or weeks)
- Burning media... lots of media
- What will legal say?
- Whose even investigating these things, anyway?



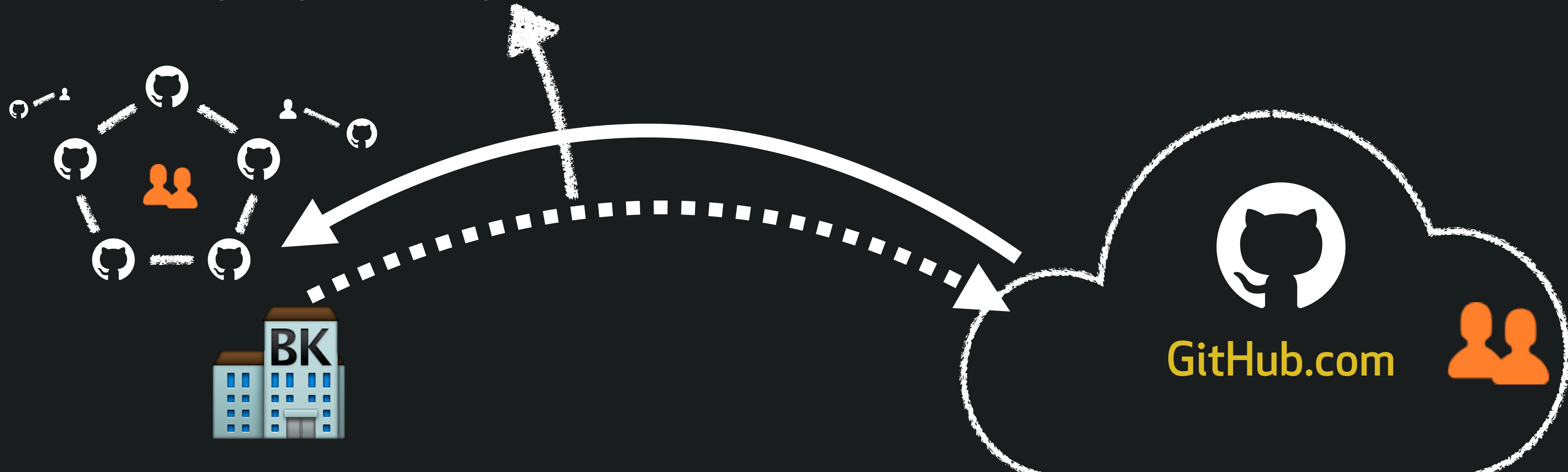
What if we used a proxy?

- Process enforcement
- Leadership Approvals
- Security
 - Pii, Dependency Checking, Dirty Words, Token Scanning, Static and Dynamic inspection, ALMSS, Vulnerability tracking
- Compliance & Audit



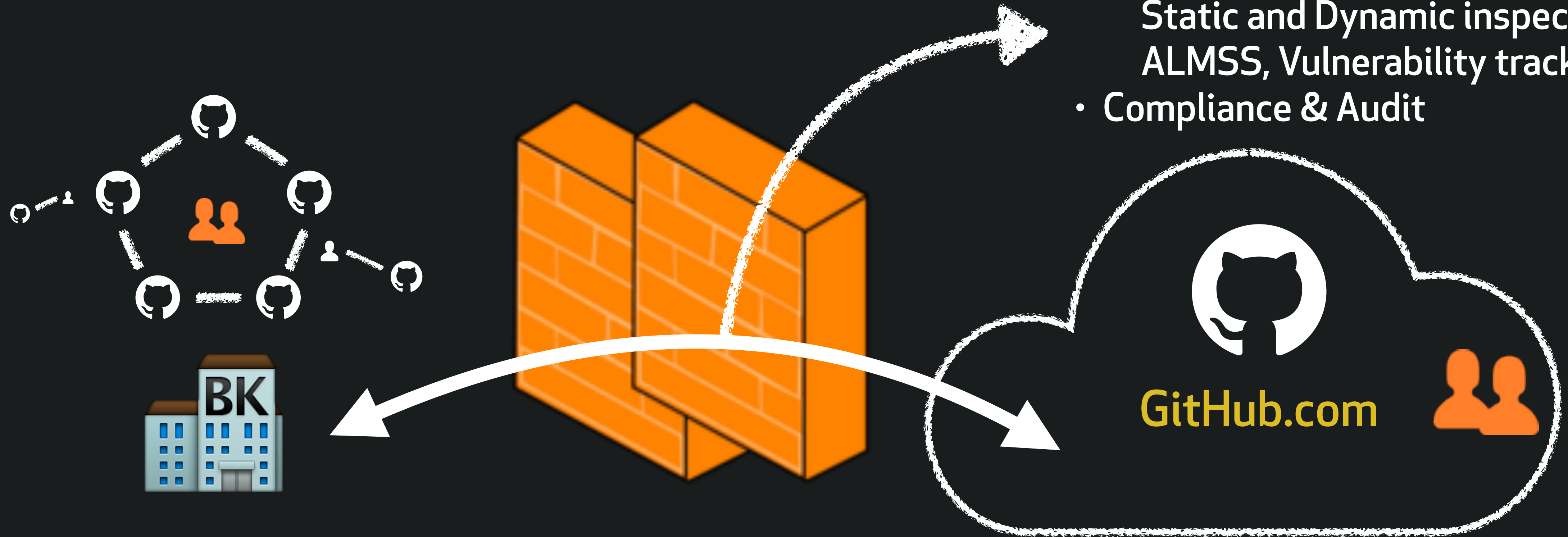
A better process? maybe?

- Filling out forms
- Getting approvals
- Waiting days (or weeks)
- Burning media... lots of media
- What will legal say?
- Whose even investigating these things, anyway?



Where we are going

- Process enforcement
- Leadership Approvals
- Security
 - Pii, Dependency Checking, Dirty Words, Token Scanning, Static and Dynamic inspection, ALMSS, Vulnerability tracking
- Compliance & Audit



Code in the open!!!!



GitHub Security Best Practices



Vulnerability Scanning

Data services

Use the data from your repository to power these enhanced features. If you'd like to enable the [dependency graph](#), vulnerability alerts, and services like it, we'll need additional permissions.

☒ **Allow GitHub to perform read-only analysis of this repository**

By checking the "Allow GitHub to perform read-only analysis of this repository" checkbox, you're agreeing to GitHub's [Terms of Service](#) and granting us permission to perform read-only analysis of this private repository. [Learn more about how we use your data.](#)

☒ **Dependency graph**

Access My_Project's dependencies, sub-dependencies, versions, and related repositories on GitHub.

☒ **Vulnerability alerts**

Receive alerts for known security vulnerabilities found in dependencies.



Alert Notification

<> Code

! Issues 0

🔗 Pull requests 0

▶ Actions

📁 Projects 0

📖 Wiki

📊 Insights

⚙ Settings

Pulse

Contributors

Traffic

Commits

Code frequency

Dependency graph

Alerts

Network

Forks

Alerts

Dismiss all ▾

⚠ 1 Open ✓ 0 Closed

Sort ▾

🟢 ! hoek

opened 9 hours ago by GitHub • package-lock.json

moderate severity

GitHub tracks known security vulnerabilities in some dependency manifest files. [Learn more about alerts.](#)



Protected Branches

ons

laborators & teams

ches

hooks

grations & services

oy keys

Branch protection rule

Apply rule to

Rule settings

Protect matching branches

Disables force-pushes to all matching branches and prevents them from being deleted.

☐ Require pull request reviews before merging

When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.

☐ Require status checks to pass before merging

Choose which [status checks](#) must pass before branches can be merged into a branch that matches this rule. When enabled, commits must first be pushed to another branch, then merged or pushed directly to a branch that matches this rule after status checks have passed.

☐ Require signed commits

Commits pushed to matching branches must have verified signatures.

☐ Include administrators

Enforce all configured restrictions for administrators.

☐ Restrict who can push to matching branches

Specify people or teams allowed to push to matching branches. Required status checks will still prevent these people from merging if the checks fail.



Required Status Checks


Enforce compliance policies with **branch statuses, branch protection, required reviews** and inline **conflict resolution**


@gromain @rodrigok is this still in the works ?

👍 2


🔧 engelgabriel modified the milestones: **0.63.0**, **0.65.0** 26 days ago

Add more commits by pushing to the **develop-markdown** branch on **gromain/Rocket.Chat**.





 **Review required** [Show all reviewers](#)


At least 1 approving review is required by reviewers with write access. [Learn more.](#)


 **Some checks haven't completed yet** [Hide all checks](#)


3 expected and 2 successful checks


 **ci/circleci: build** *Expected — Waiting for status to be reported* **Required**

 **ci/circleci: test-with-oplog** *Expected — Waiting for status to be reported* **Required**

 **ci/circleci: test-without-oplog** *Expected — Waiting for status to be reported* **Required**

 **continuous-integration/travis-ci/pr** — The Travis CI build passed [Details](#)

 **license/ccla** — Contributor License Agreement is signed. **Required** [Details](#)

 **This branch has conflicts that must be resolved** [Resolve conflicts](#)

Use the [web editor](#) or the [command line](#) to resolve conflicts.

Conflicting files

package.json

packages/rocketchat-markdown/markdown.js

packages/rocketchat-ui-message/client/message.html

packages/rocketchat-ui-message/client/renderMessageBody.js

Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).




Restrict access


Enforce all configured restrictions for administrators.


☒ **Restrict who can push to matching branches**
Specify people or teams allowed to push to matching branches. Required status checks will still prevent these people from merging if the checks fail.

🔍 jbjon


 **jbjonesjr** Jamie Jones

People and teams with push access

 **Organization and repository administrators**
These members can always push.

 **hubot**
Hubot

✕



Code Owners

```
> cat CODEOWNERS

# Lines starting with '#' are comments.
# Each line is a file pattern followed by one or more owners.

# These owners will be the default owners for everything in the repo.
*      @defunkt

# Order is important. The last matching pattern has the most precedence.
# So if a pull request only touches javascript files, only these owners
# will be requested to review.
*.js    @octocat @github/js

# You can also use email addresses if you prefer.
docs/*   docs@example.com
```



Cleanup after yourself

Danger Zone

Make this repository public

Make this repository visible to anyone.

Make public

Transfer ownership

Transfer this repository to another user or to an organization where you have the ability to create repositories.

Transfer

Archive this repository

Mark this repository as archived and read-only.

Archive this repository

Delete this repository

Once you delete a repository, there is no going back. Please be certain.

Delete this repository



Bonus: signed commits for additional assurance

Rule settings
Protect matching branches Disables force-pushes to all matching branches and prevents them from being deleted.
<input type="checkbox"/> Require pull request reviews before merging When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.
<input type="checkbox"/> Require status checks to pass before merging Choose which status checks must pass before branches can be merged into a branch that matches this rule. When enabled, commits must first be pushed to another branch, then merged or pushed directly to a branch that matches this rule after status checks have passed.
<input checked="" type="checkbox"/> Require signed commits Commits pushed to matching branches must have verified signatures.
<input type="checkbox"/> Include administrators Enforce all configured restrictions for administrators.
<input type="checkbox"/> Restrict who can push to matching branches Specify people or teams allowed to push to matching branches. Required status checks will still prevent these people from merging if the checks fail.



Open Source Security & Compliance Best Practices



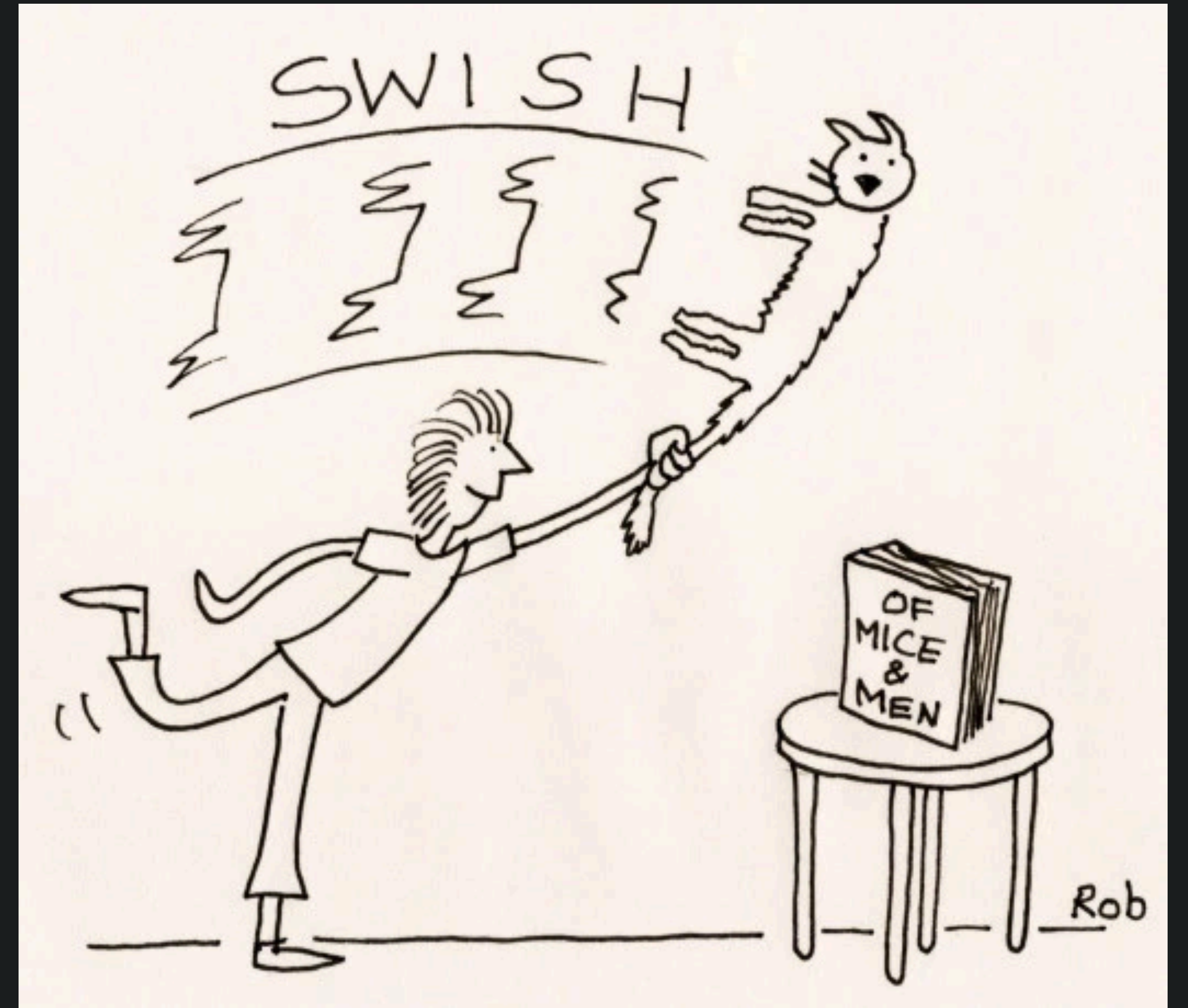
Dependency Management

- Dependency Management tools like
 - **greenkeeper.io, Dependabot, Depfu, Renovate**
- ...Or check nightly via your own custom scripts
 - Download, run dependency update per language spec, use diff to create patch



Container and image scanning

- You can't swing a dead cat without hearing about scanning solutions:
 - **Kritis, Clair, AquaSec, Sonatype, BlackDuck, OpenSCAP**
- Important to remember that container scanning is not **THE** security process, but a piece of it

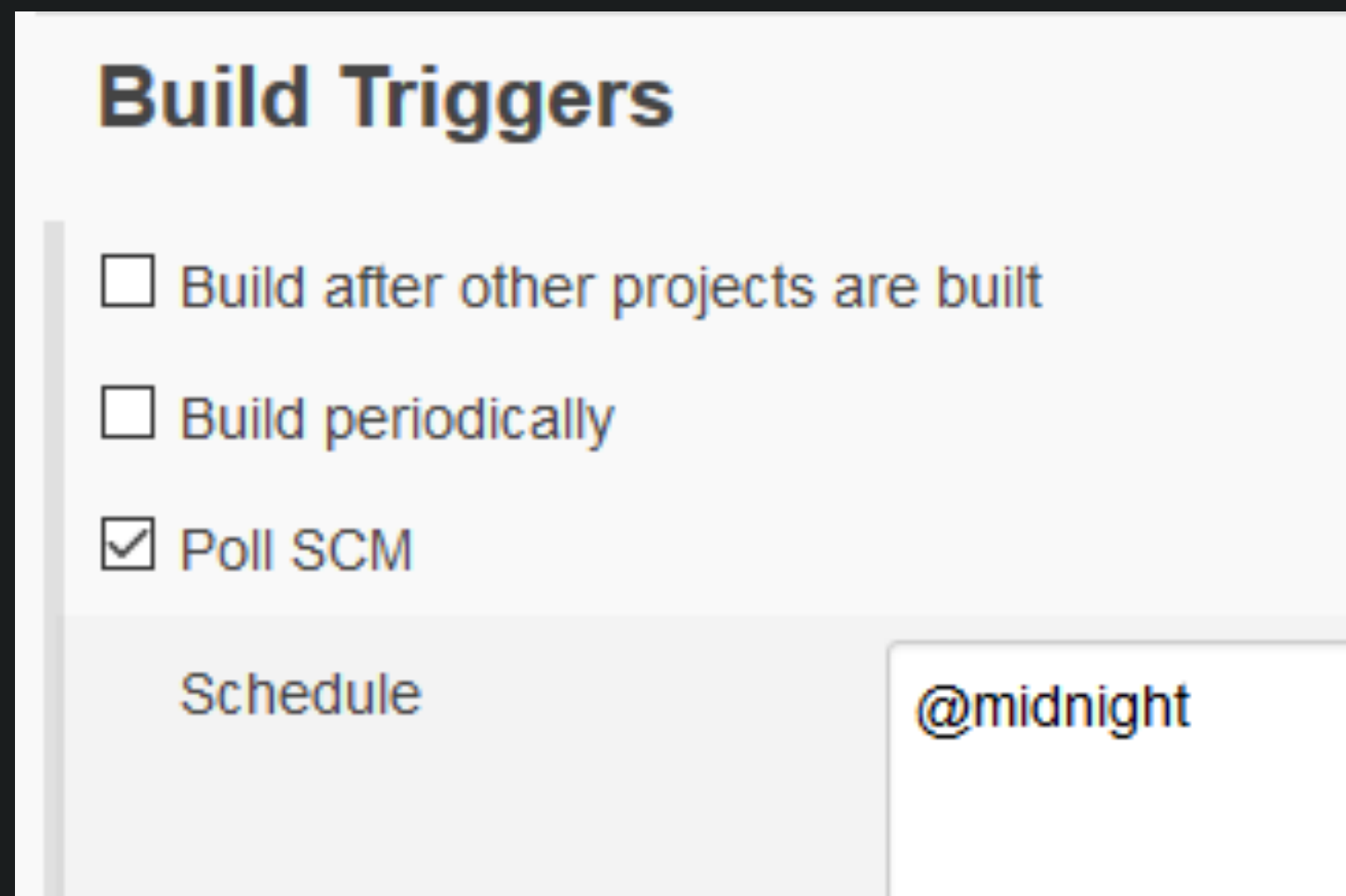


Vulnerability data

- While GitHub includes data from the NVD system and our own platform research, there are numerous other providers of data
- Snyk
- Whitesource
- BlackDuck
- Sonatype
- All leading to... your OWN threat intelligence



Build often



The screenshot shows a 'Build Triggers' configuration panel. It has three checkboxes: 'Build after other projects are built' (unchecked), 'Build periodically' (unchecked), and 'Poll SCM' (checked). Below these is a 'Schedule' field with the value '@midnight'.

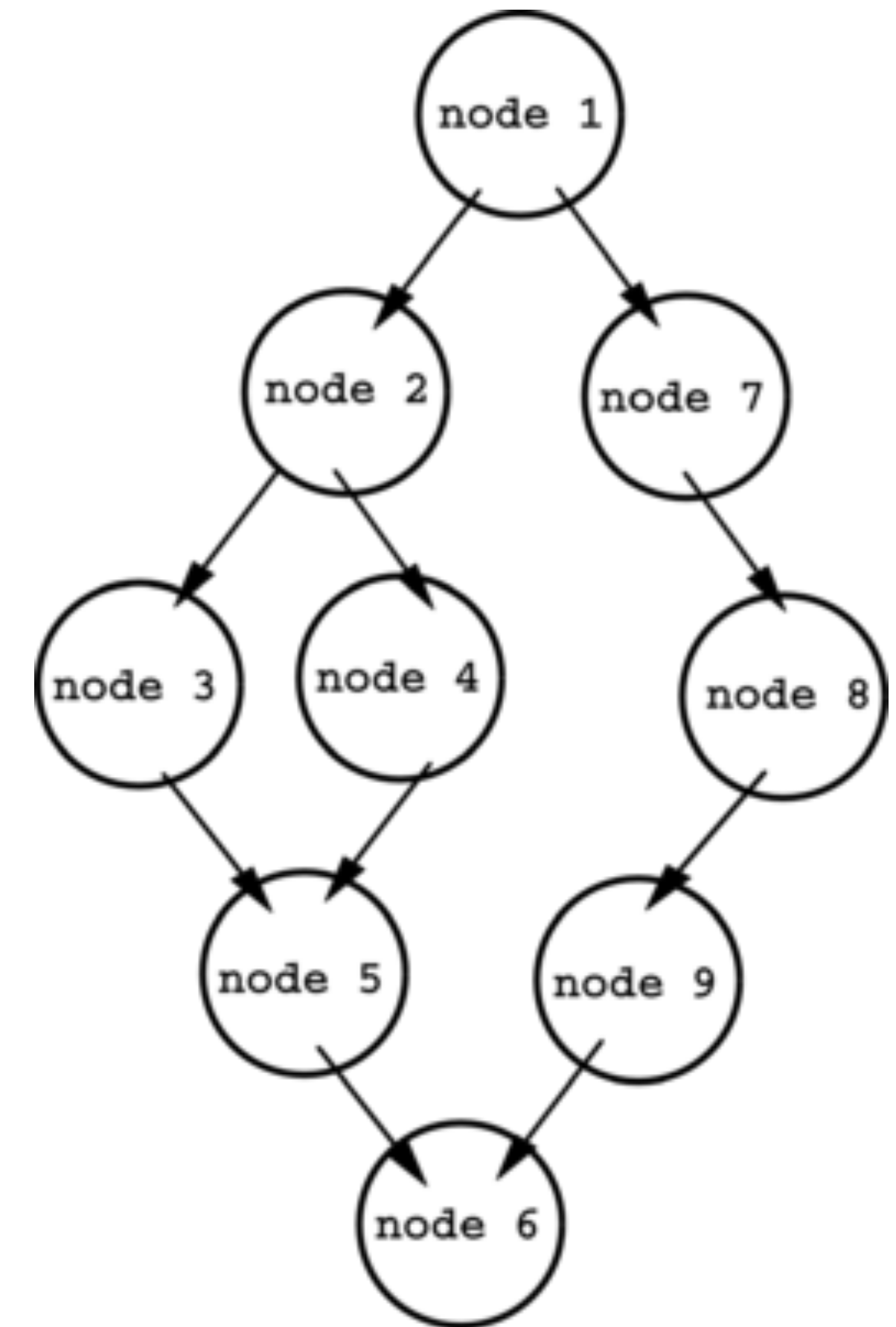
Build Triggers
<input type="checkbox"/> Build after other projects are built
<input type="checkbox"/> Build periodically
<input checked="" type="checkbox"/> Poll SCM
Schedule
@midnight

- Don't just run builds after commits or merges, but on routine basis as well!
- You never know when your bugs will appear



Static, Dynamic, and Browser Testing! Oh My!

- Static Code Analysis Tools like
 - SonarQube, Codacy, WhiteSource, OWASP, Fortify, Veracode, Coverity
- Dynamic Code Analysis Tools like
 - Veracode, Selenium, OWASP ZAP



Document and Security and Compliance policy

- We talk about what licenses we allow, but do we plan for the vulnerabilities we allow?
- What about the flavor of contributor we encourage?
- Are their activity requirements on the dependencies? Do they have to be active at all?



Open Source tools and Integrations



Vulnerability tools

Whitesource, Veracode, Sonatype, IonChannel, GitHub

Static Code Analysis

SonarQube, Selenium, Fortify, Codacy, OWASP, Coverity

Dynamic Code Analysis

Veracode, Selenium, OWASP ZAP

Dependency Management

Dependabot, greenkeeper.io, Renovate, Depfu

Container Scanning

Kritis, Clair, AquaSec, Sonatype, BlackDuck, OpenSCAP

Vulnerability Resources

Whitesource, Sonatype, Snyk, BlackDuck



Open Developer Platform

Hosted Platforms

Future partnerships and contributions

SYMPHONY
(ReST API)

SYMPHONY
(Extension API)

SYMPHONY
(Integration webhooks)

FINTECH
OPEN APIS

CLOUD
OPEN APIS

FINTECH
OPEN DATA

High Productivity Turnkey Developer Experience

Development Infrastructure



Biz & Legal Peace Of Mind - We Do The Hard Part!

Collaboration Services

WIKI

 Atlassian Confluence

MAILING LISTS

 Google Groups

WEB CONFERENCING

 WebEx

METRICS & REPORTING

 Bitergia

Recap

- Understanding the different workflows used publishing code outside your organization
- Understanding GitHub and integration points key for process security
- Better practices for open source security
- Things to think about when staying legal and compliant with open source code
- Critical tools to integrate into your process



Code Thoughts™ for Open Source participation



Code Thoughts™ v1

- How are you managing a community?
- Where do they exist today? How are you going to meet them there?
- What happens when someone acts inappropriately? What's your PR strategy?
- Everything (EVERYTHING) in the Core Infrastructure Initiative Best Practice program



Code Thoughts™ v2 - With more security

- Who secures your projects?
 - Your internal security team? `The community`?
Who is responsible for those critical fixes? But what if YOU'RE not there?
- Do you even audit?
- Do you have an incident response plan for severe vulnerabilities?
 - Where do you report? Do you?
 - https://github.com/envoyproxy/envoy/blob/master/SECURITY_RELEASE_PROCESS.md



 github.com/jbjonesjr/open-source-docs

**finosfoundation.atlassian.net/wiki/spaces/
FDX**

  @jbjonersjr

jbjonersjr@github.com

Questions and Answers?

