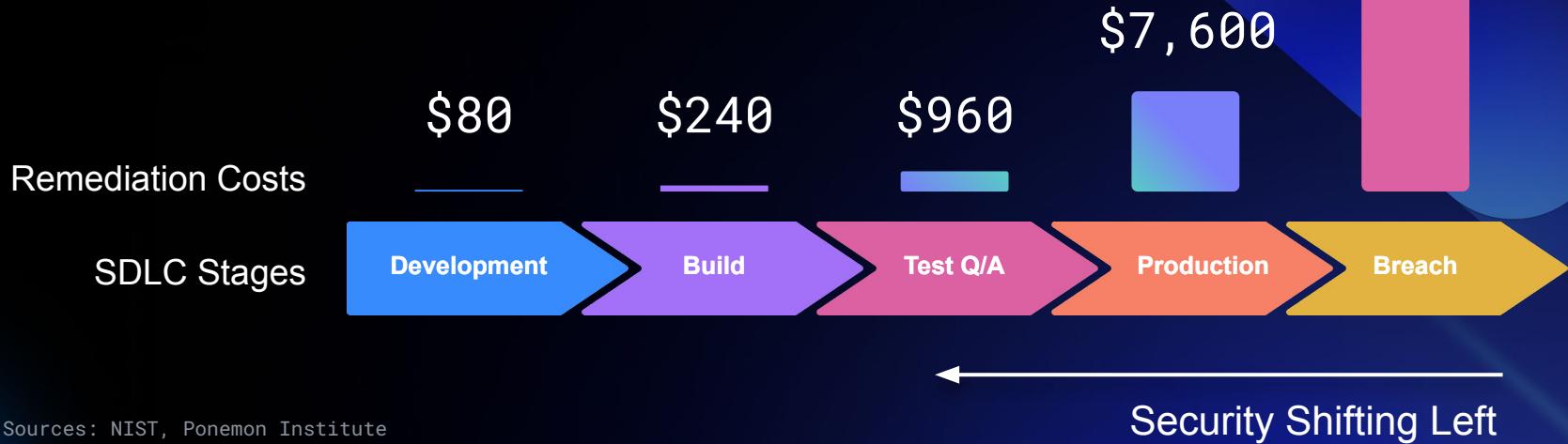


Shift left... the right way

Christopher Harrison
@geektrainer
Senior Developer Advocate



Everyone wants to shift security left...



Sources: NIST, Ponemon Institute

Security debt is pervasive

Security debt is defined as all security flaws and vulnerabilities that remain unremediated for over one year

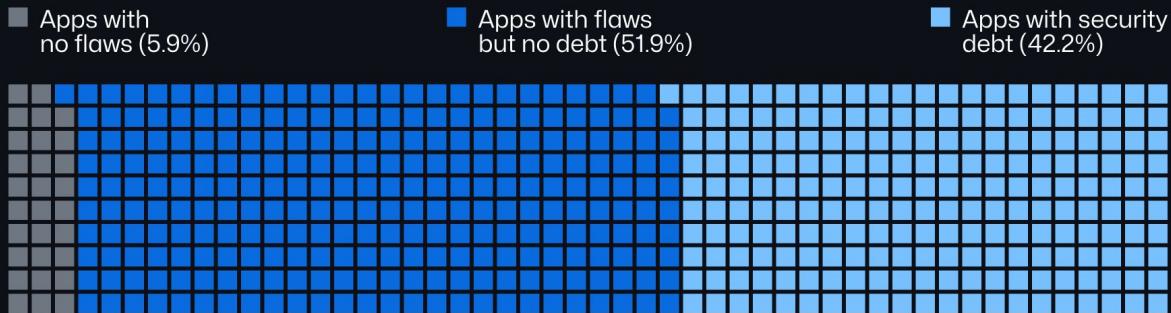


Figure 8: Prevalence of security debt across all applications greater than one year old

Source: Veracode SOSS 2024

Flaws but no debt: Flaws exist in security backlog but <1 year old
Security debt: Flaws and vulnerabilities >1 year old

70.8%

of organizations have security debt

89.4%

of security debt exists in first party code

46.6%

of security flaws in the median organization will be Security Debt

Agenda

How we currently do things

How we should do things

How we put things into action

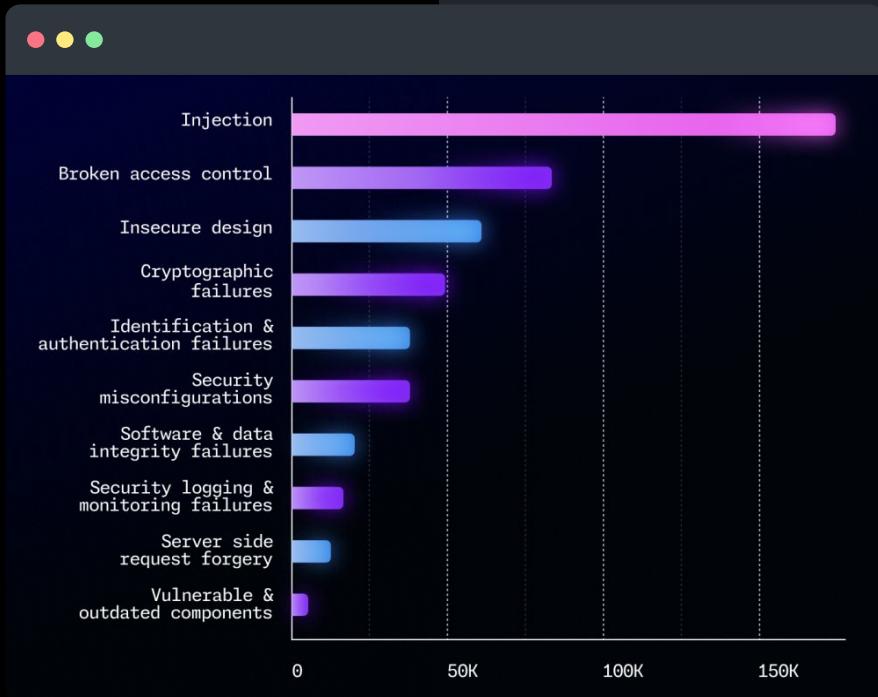
How we currently do things

Make developers
security professionals

The top weaknesses are not new

Rank		Weakness Description	CWE ID	CVEs in KEV	Rank Last Year	Change
1		Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	CWE-79	3	2 (up 1)	▲
2		Out-of-bounds Write	CWE-787	18	1 (down 1)	▼
3		Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	CWE-89	4	3	
4		Cross-Site Request Forgery (CSRF)	CWE-352	0	9 (up 5)	▲
5		Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	CWE-22	4	8 (up 3)	▲
6		Out-of-bounds Read	CWE-125	3	7 (up 1)	▲
7		Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	CWE-78	5	5 (down 2)	▼
8		Use After Free	CWE-416	5	4 (down 4)	▼
9		Missing Authorization				

Most common types of vulnerabilities found by CodeQL in 2024



Most common vulnerability, per language

GitHub public repos
Detected by Code Scanning
Aug. 2023 - Aug. 2024



JavaScript

Incomplete sanitization (XSS)

34k alerts



Python

Clear-text logging of sensitive data

32k alerts



Java

Cross-site scripting (XSS)

21k alerts



C/C++

Unsafe type casting

20k alerts



Go

Incorrect integer conversion

8k alerts



C#

Execution of arbitrary system commands

3k alerts

A screenshot of a web browser displaying the BleepingComputer website. The page title is "Hackers steal data of 2 million in SQL injection, XSS attacks". The main image shows a laptop keyboard with a glowing red hand reaching out from the screen, symbolizing cybercrime. The article summary states that a threat group named 'ResumeLooters' has stolen personal data from over two million job seekers by compromising 65 legitimate job listing and retail sites using SQL injection and cross-site scripting (XSS) attacks.

BLEEPINGCOMPUTER

NEWS ▾ TUTORIALS ▾ VIRUS REMOVAL GUIDES ▾ DOWNLOADS ▾ DEALS ▾ VPNs ▾ FORUMS ▾ MORE ▾

Home > News > Security > Hackers steal data of 2 million in SQL injection, XSS attacks

By Bill Toulias February 6, 2024 02:00 AM 1

Hackers steal data of 2 million in SQL injection, XSS attacks

A threat group named 'ResumeLooters' has stolen the personal data of over two million job seekers after compromising 65 legitimate job listing and retail sites using SQL injection and cross-site scripting (XSS) attacks.

POPULAR STORIES

Zero-click Windows TCP/IP RCE impacts all systems with IPv6 enabled, patch now

Microsoft retires Windows updates causing 0x80070643 errors

Why does this still happen?

Developers are taught
insecure methods
Often tutorials bypass proper
security

Difficult to see the impact
of secure code
There's no red/green process
like testing

Security training isn't
impactful
"How does this impact me?"

Well-meaning developers
take shortcuts
"I just need this to work. I'll
fix it later."

Hire security teams!

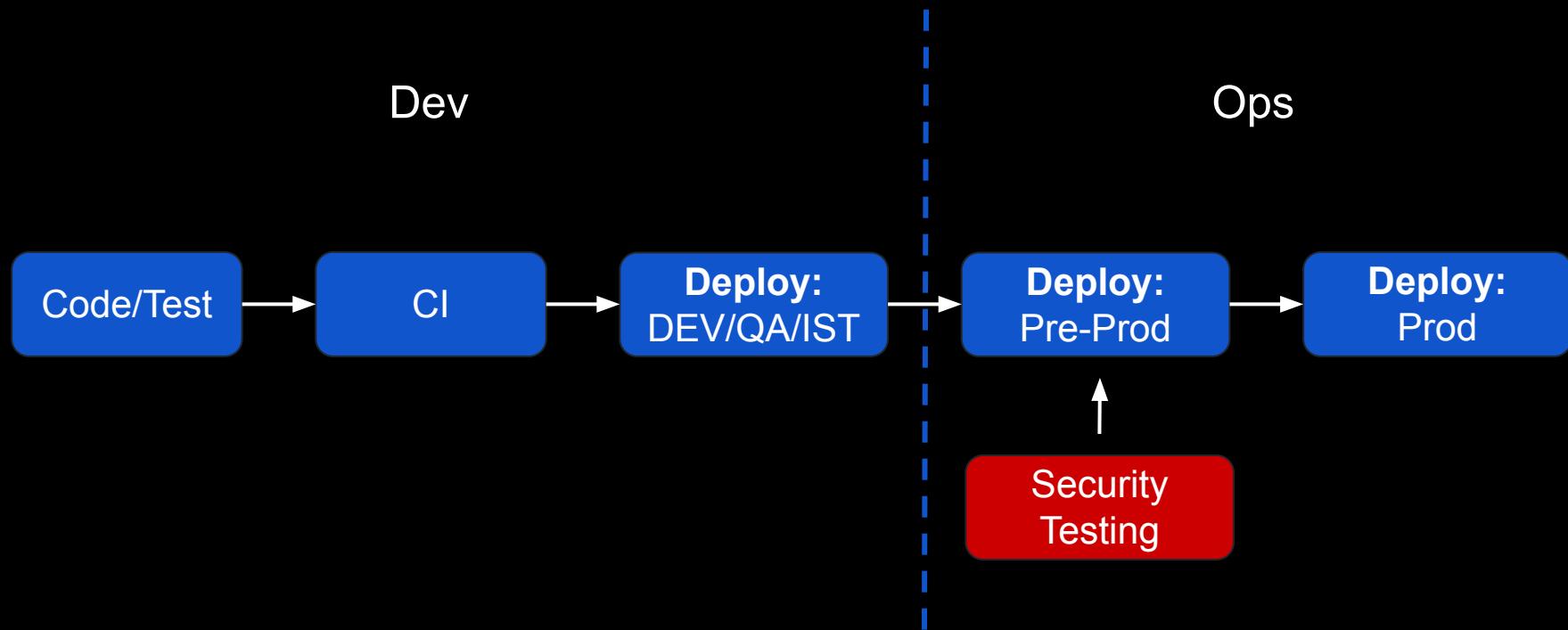
The Dev-Sec Divide

Developers

Security



DevOps Pipeline



Separate teams with (what feels like) separate missions

Always outnumbered
(and overworked)

Traditionally involved late
in the process

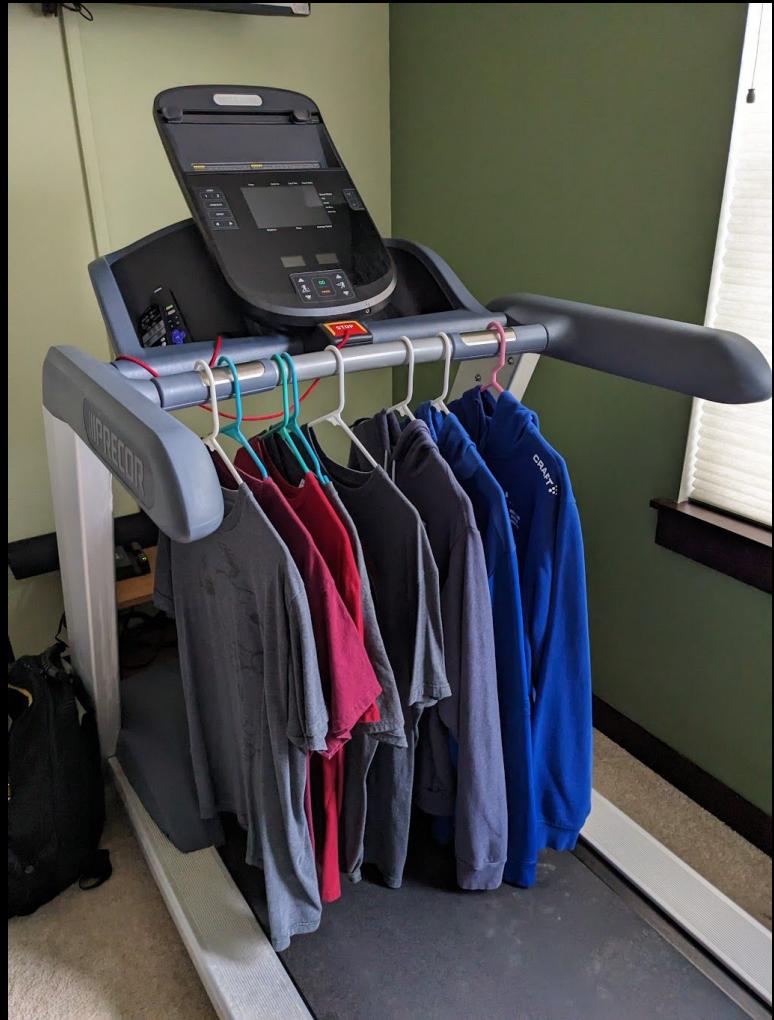
Unfairly criticized for
“misaligned objectives”

BUY MORE TOOLS!!

“

**Security at the expense of usability
comes at the expense of security.**

Avi Douglen
OWASP Board of Directors



New tools don't magically deliver new culture

Shiny objects are quickly
forgotten

Wrong tools for the job

Wrong tools for the
culture

How we should do things

But first...
Let's define
DevOps and
DevSecOps

Key focus areas

Assume best intent
Nobody is checking in unsecure code on purpose

Review the output
Give feedback on the code and the process, not the humans involved

We're a team
Everyone has different roles and strengths; let's play to those

Automate automation
Streamline the process as much as possible

**How do we do the
thing?**

Take a swiss
cheese approach

The code we write

**A moment for our
overloaded devs**

**Remember:
good DevOps is key to
good DevSecOps**

Supporting developers

Code smells

Teach developers to involve security early when they're working on something which "smells" interesting

Documentation

Devs want to do the right thing; they just don't know how

Inline upskilling

Show developers on **their code** where problems arise and how to best avoid them

Team work

We're all on the same team with the same goals

The role of GenAI



How GenAI can help developers

Explain code

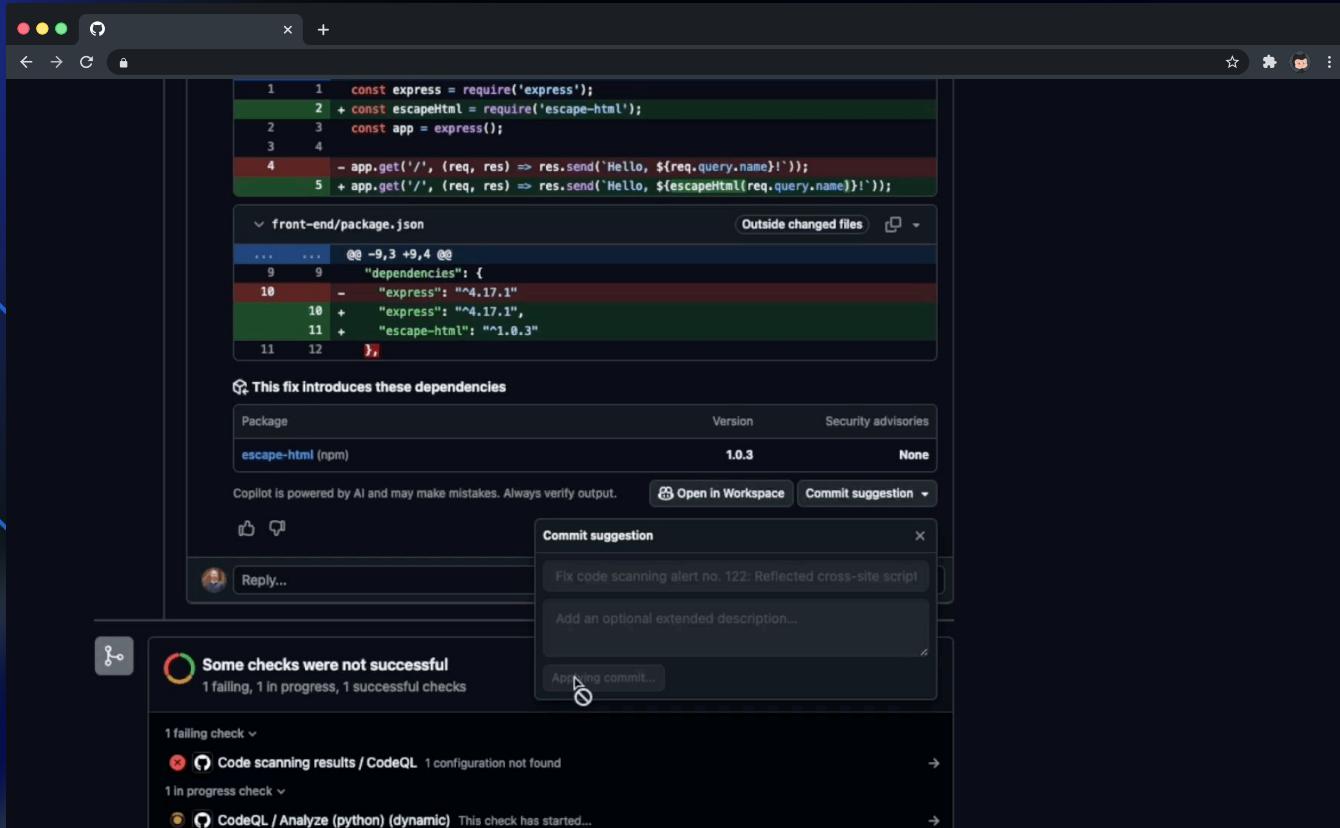
Ask your assistant if there are any potential issues and how to resolve them

Create code

AI assistants can generate secure code either by design or as prompted

**What if the code is
already written?**

Autofix in the pull request



Autofix for existing alerts

The screenshot shows a GitHub repository page for 'beat-bot-backend'. The repository was generated from [octodemo/beat-bot](#). The commit history lists 1 branch and 1 commit by user 'leftrighthand'. The commits are all initial commits for files like .github/workflows, front-end, src, test, .env, .gitignore, Dockerfile, README.md, credentials.txt, entrypoint.sh, and requirements.txt, all made 2 weeks ago. The repository has 0 stars, 0 forks, and 0 releases. It uses Python 96.5% of the codebase, with Dockerfile at 2.1% and Other at 1.4%.

Code | **Issues** | **Pull requests** | **Actions** | **Projects** | **Wiki** | **Security** 9 | **Insights** | **Settings**

beat-bot-backend Internal
generated from [octodemo/beat-bot](#)

main 1 Branch Tags

leftrighthand Initial commit · 5d464d7 · 2 weeks ago 1 Commit

.github/workflows Initial commit 2 weeks ago

front-end Initial commit 2 weeks ago

src Initial commit 2 weeks ago

test Initial commit 2 weeks ago

.env Initial commit 2 weeks ago

.gitignore Initial commit 2 weeks ago

Dockerfile Initial commit 2 weeks ago

README.md Initial commit 2 weeks ago

credentials.txt Initial commit 2 weeks ago

entrypoint.sh Initial commit 2 weeks ago

requirements.txt Initial commit 2 weeks ago

About

No description, website, or topics provided.

Readme Activity Custom properties 0 stars 0 watching 0 forks

Releases

No releases published [Create a new release](#)

Packages 1

best-bot-demo-2

Languages

Python 96.5% Dockerfile 2.1% Other 1.4%

Campaigns

Define the highest-risk problems
Security teams define the subset of vulnerabilities to resolve

Track the resolution of issues
Visibility into the state of remediation

The screenshot shows a web browser window with a dark theme. The title bar says "XSS managed by leftrightleft". The main content area displays a "Campaign progress" bar at 25% (4 alerts) with 12 alerts left. A status box shows "65 days left" due by Fri, Jan 31, 2025. Below this, a table lists 12 open alerts under the heading "is:open". Each alert entry includes a summary, a "High" priority badge, and a link to the issue details. To the right, a sidebar for "Copilot Autofix" shows "16 supported" alerts with a progress bar at 33% closed (12).

Open	Closed
12	4

Filter: is:open

Open 12 Closed 4

▼ android-demo

⌚ Reflected cross-site scripting (High)
#31 · Opened last year · Detected by CodeQL in server/node_backend/index.js:443

⌚ Reflected cross-site scripting (High)
#29 · Opened last year · Detected by CodeQL in server/node_backend/index.js:354

⌚ Reflected cross-site scripting (High)
#28 · Opened last year · Detected by CodeQL in server/node_backend/index.js:332

⌚ Reflected cross-site scripting (High)
#27 · Opened last year · Detected by CodeQL in server/node_backend/index.js:327

⌚ Reflected cross-site scripting (High)
#26 · Opened last year · Detected by CodeQL in server/node_backend/index.js:289

⌚ Reflected cross-site scripting (High)
#25 · Opened last year · Detected by CodeQL in server/node_backend/index.js:263

⌚ Reflected cross-site scripting (High)
#24 · Opened last year · Detected by CodeQL in server/node_backend/index.js:179

⌚ Reflected cross-site scripting (High)
#23 · Opened last year · Detected by CodeQL in server/node_backend/index.js:160

Copilot Autofix
16 supported
33% closed (12)

The code we use

90%

of code in new
applications is
open source



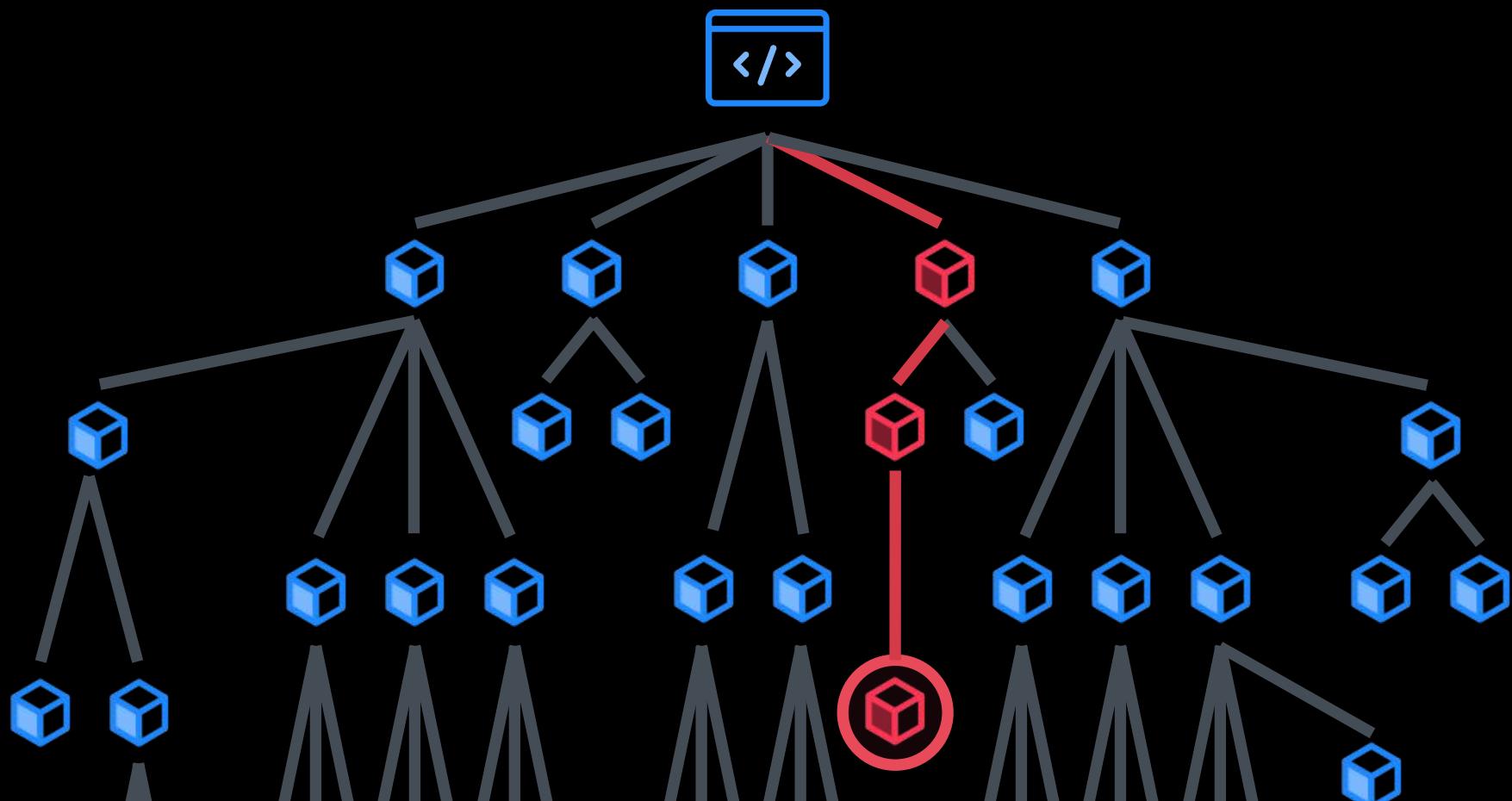


2,238 direct contributors

7,413 3rd degree contributors

17,129 5th degree contributors

23,266 7th degree contributors



Staying current

“Bump the version”

Only possible if you have good DevOps processes

Time to rewrite?

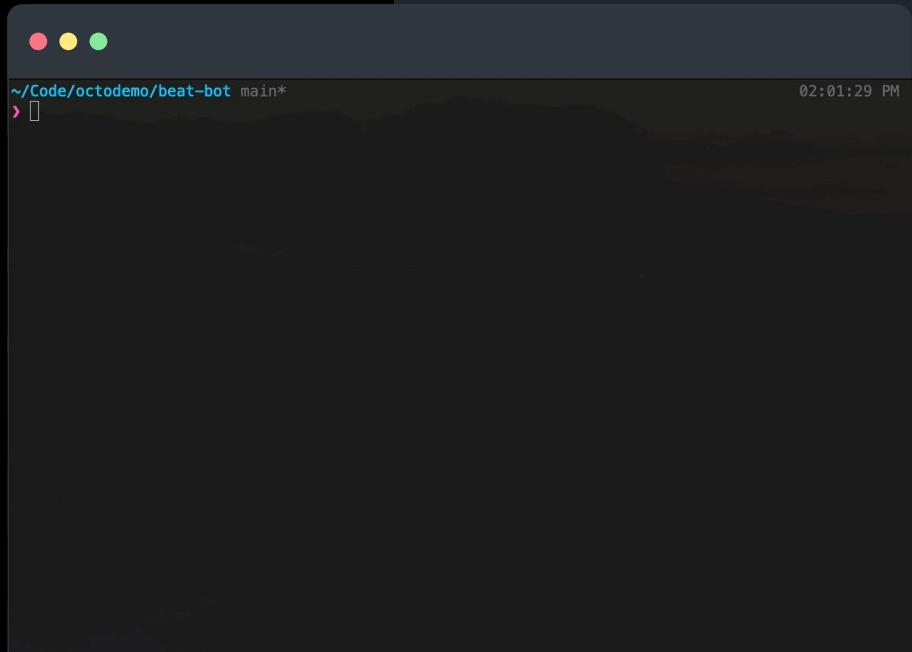
Longer the wait, the more expensive the fix

Keys in code

Pop quiz:
How many times per
month are AWS
credentials written to
public GitHub repos?

Scan for secrets and block on push

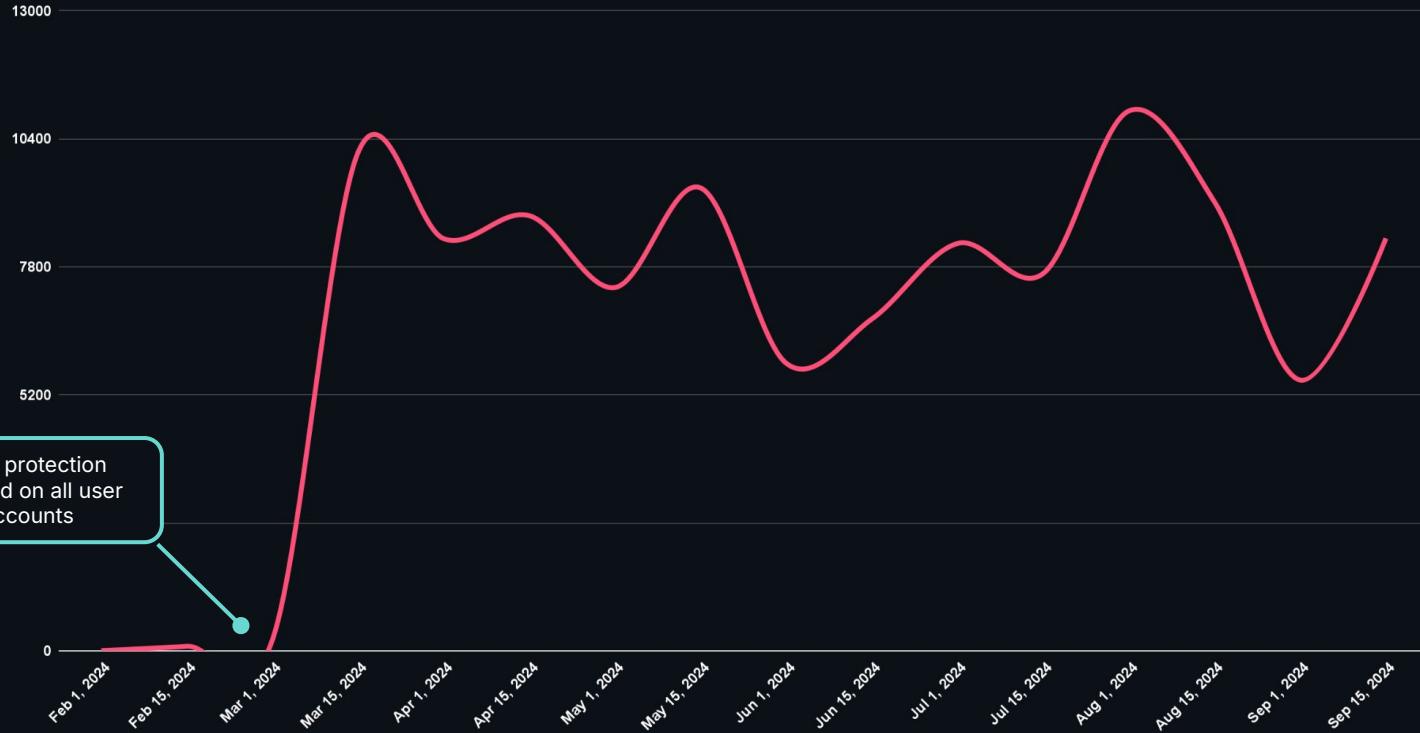
- Block new credentials from being written to GitHub
- Provides a bypass mechanism



```
~/Code/octodemo/beat-bot main*
>
```

Push Protection Blocks

GitHub public repos, Feb. 2024 - Aug. 2024



Push protection
enabled on all user
accounts

Credential considerations

Key management

How easy is it for your devs to do the right thing?

Mitigation plan

What's your plan for rotating and removing keys?

Where do we go
from here?

Use the right tools

Play to people's
strengths

Have empathy



Thank you

