Netnod Tech Meeting 2026

October 6, 2026

Daniel Stenberg



High volume AI vulnerability reports

Daniel Stenberg

@bagder@mastodon.social







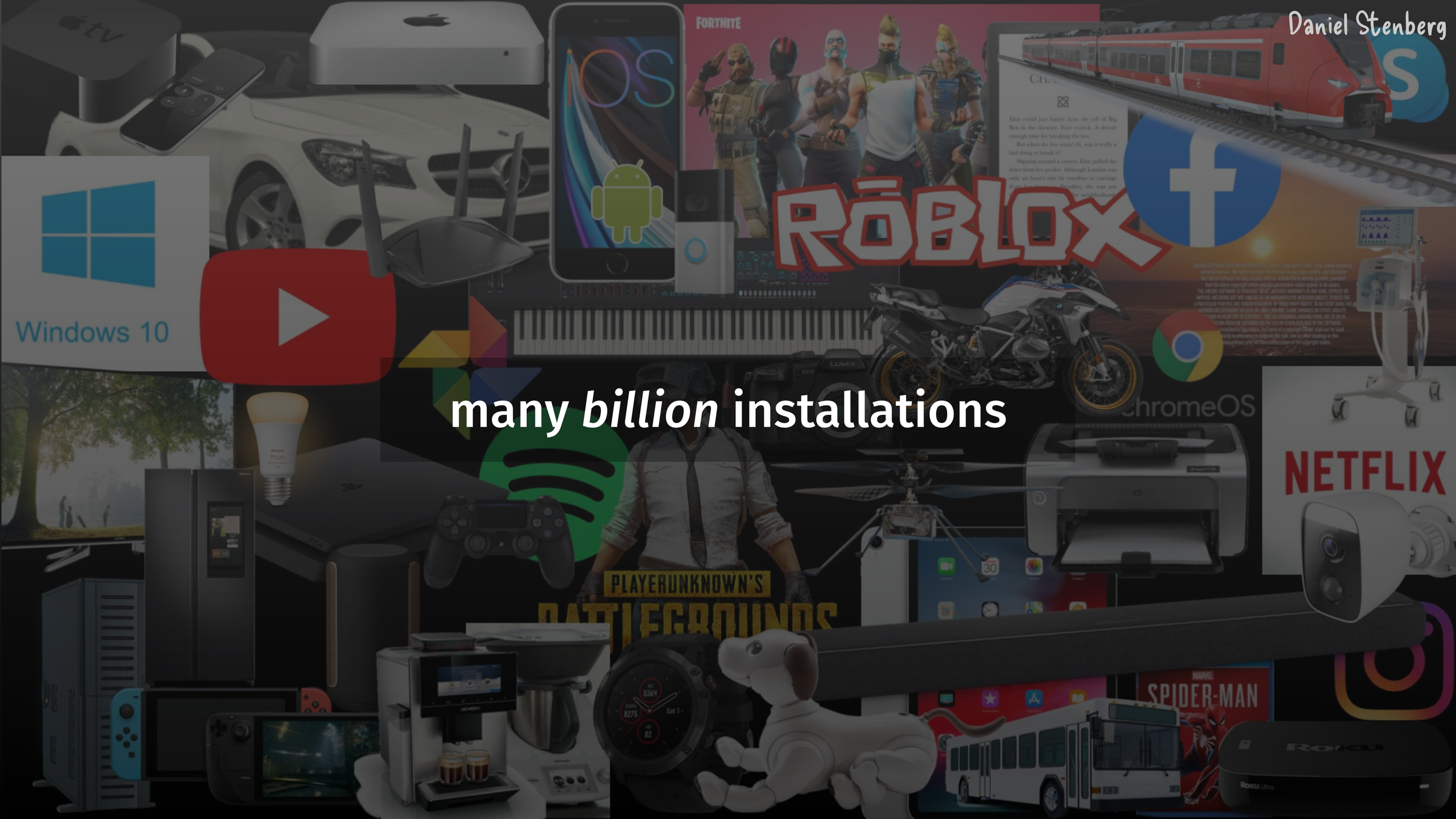
European Open Source Academy

wolfSSL

https://daniel.haxx.se

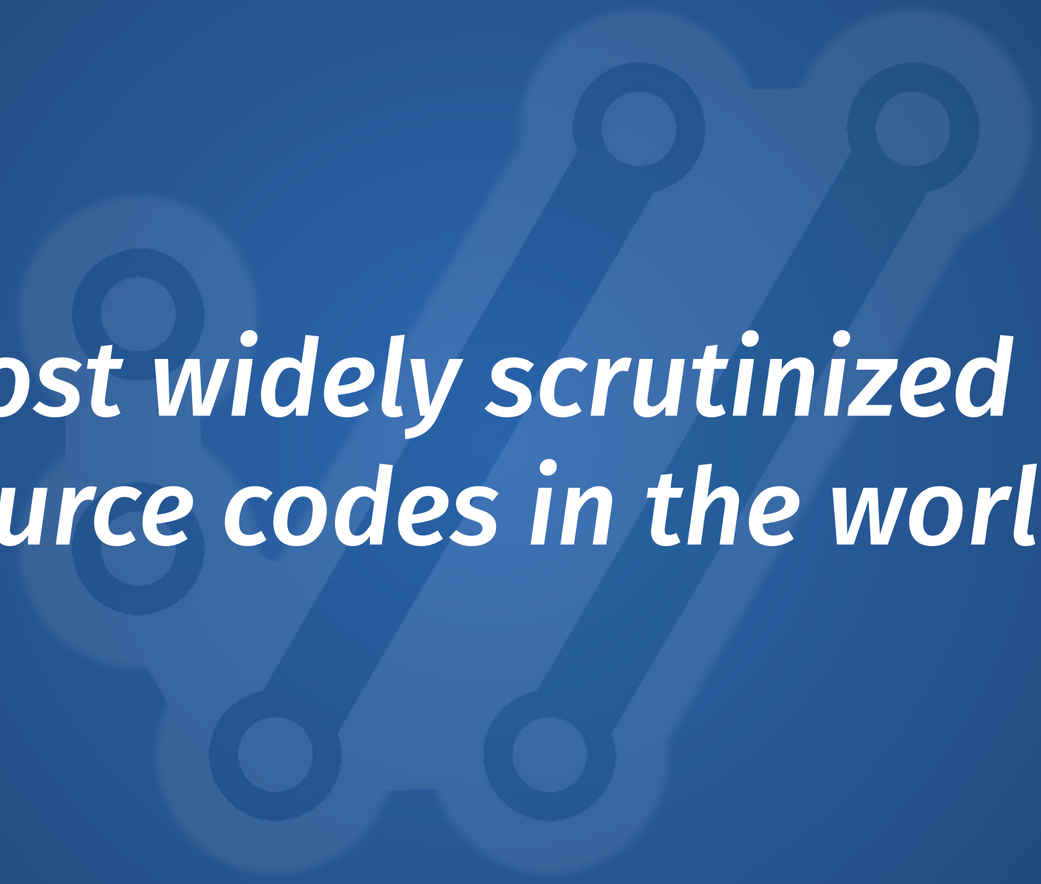
Daniel Stenberg



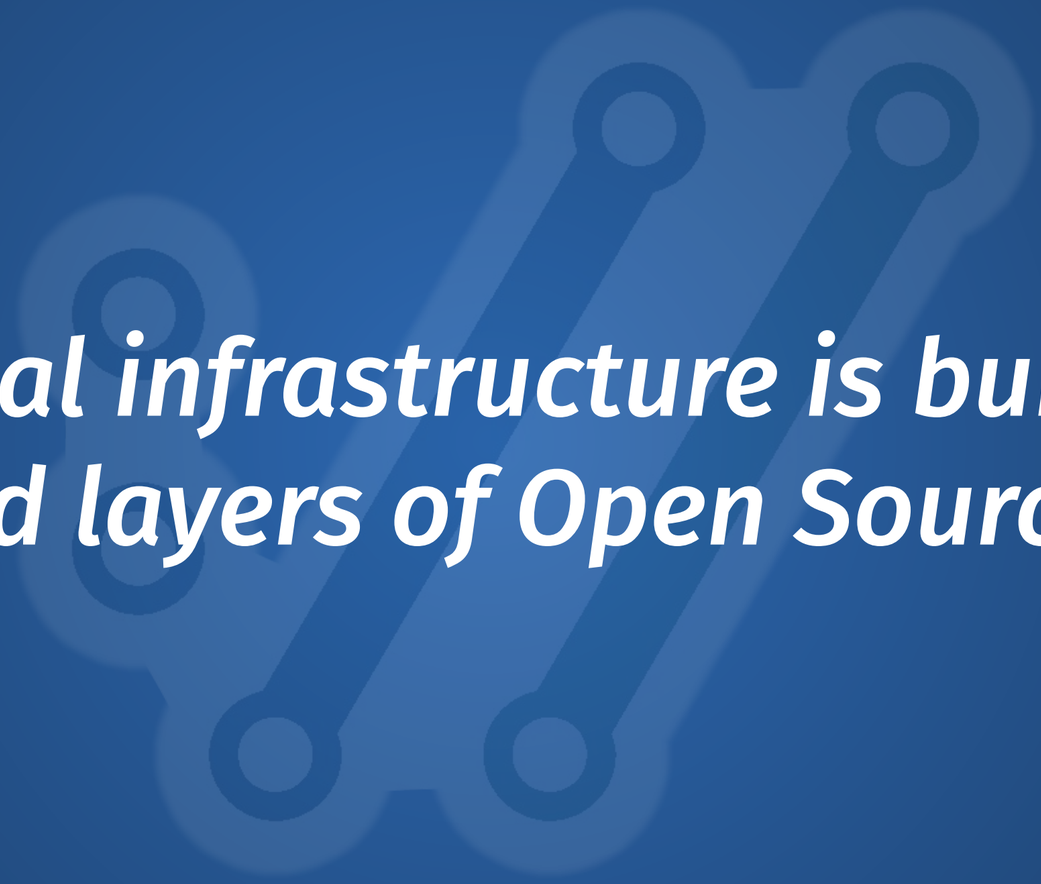


many *billion* installations

Daniel Stenberg

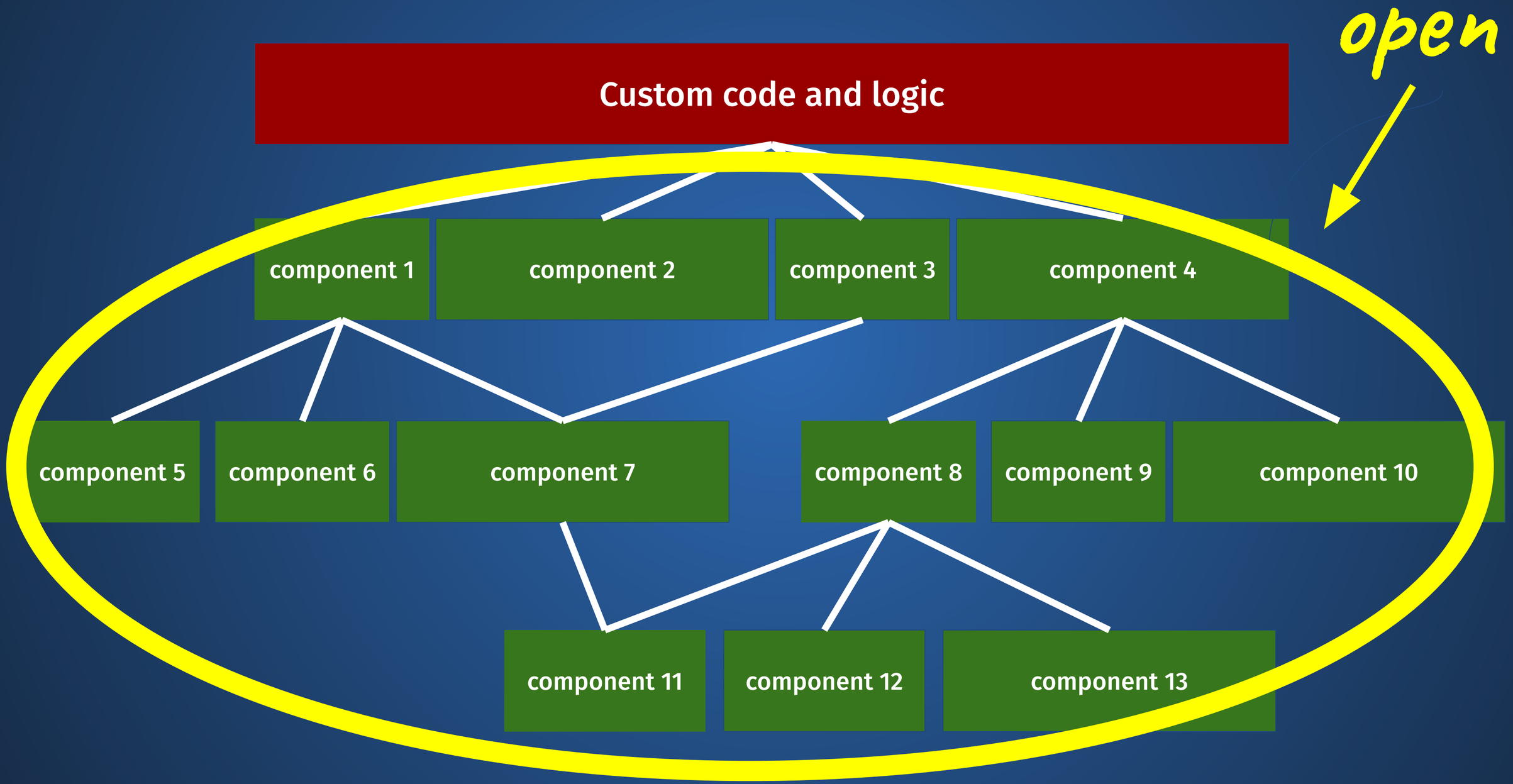


One of the most widely scrutinized and audited source codes in the world

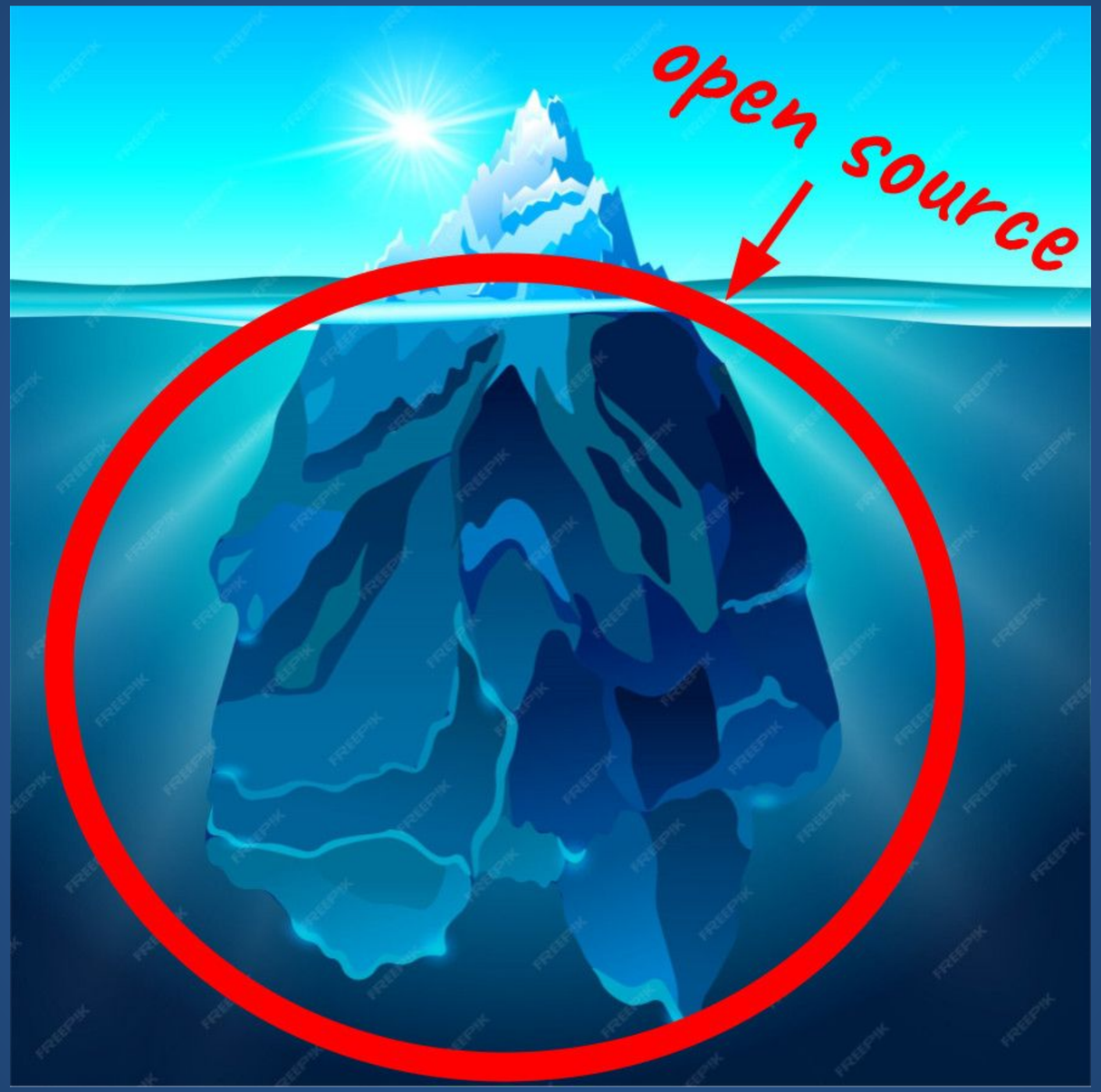


Modern digital infrastructure is built on layers and layers of Open Source

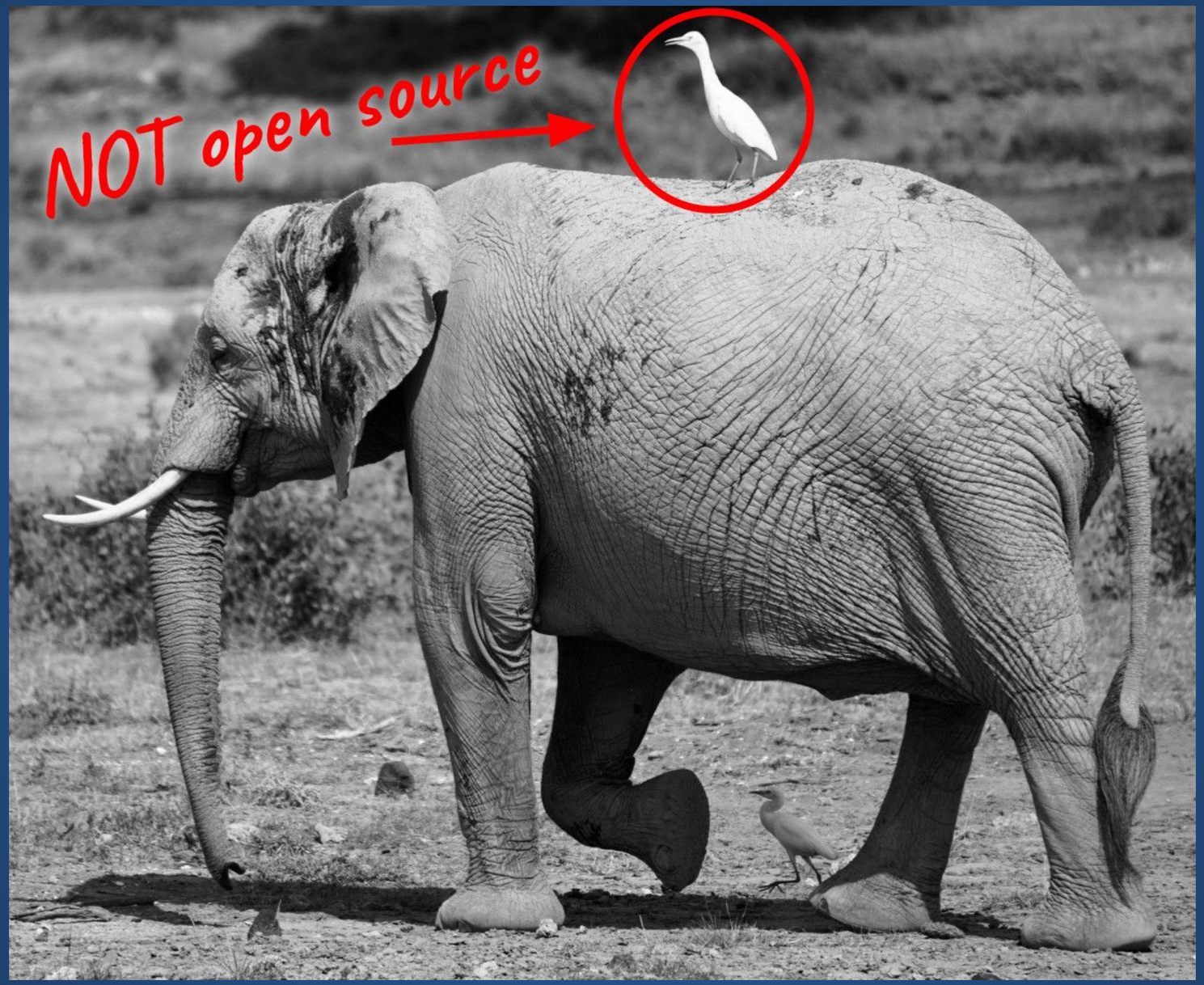
All modern software

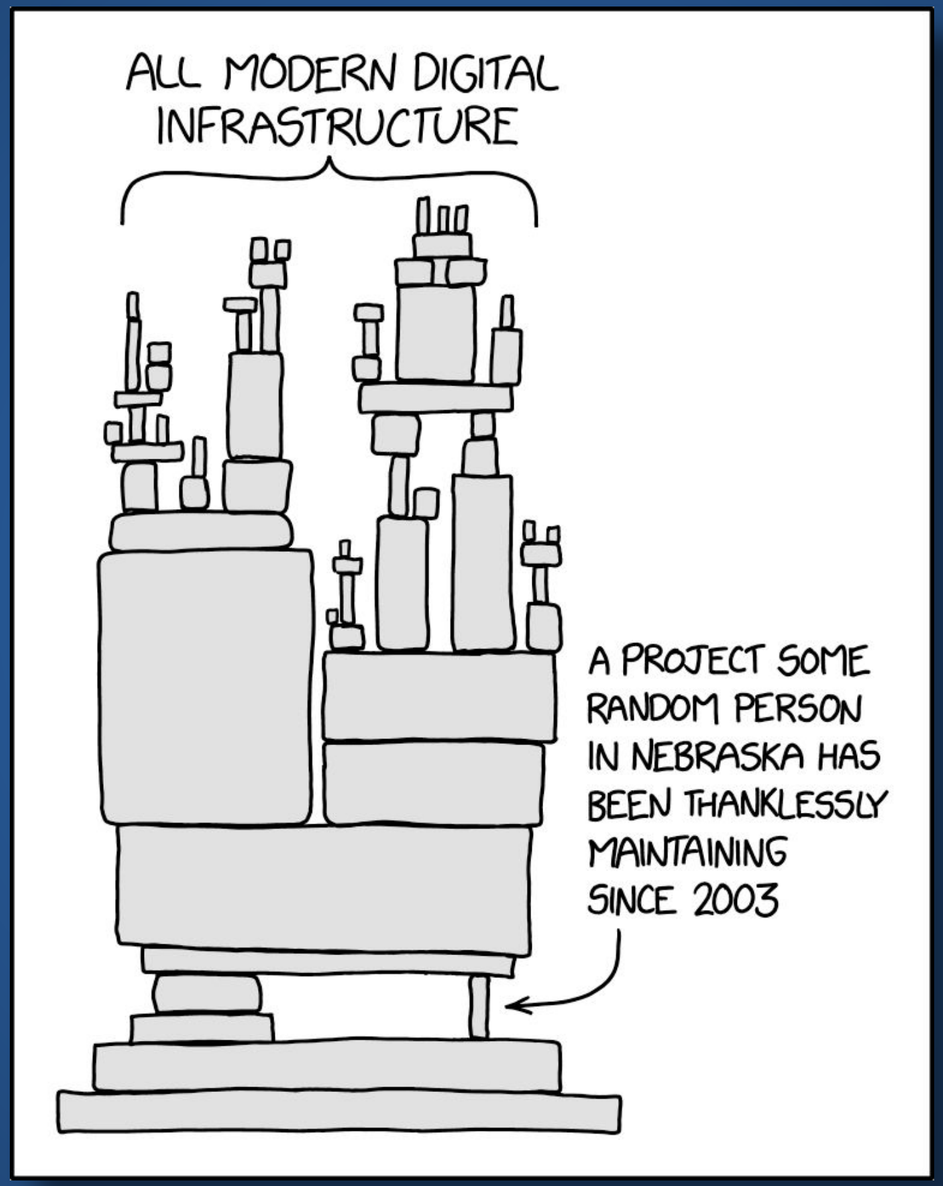


Daniel Stenberg



Daniel Stenberg

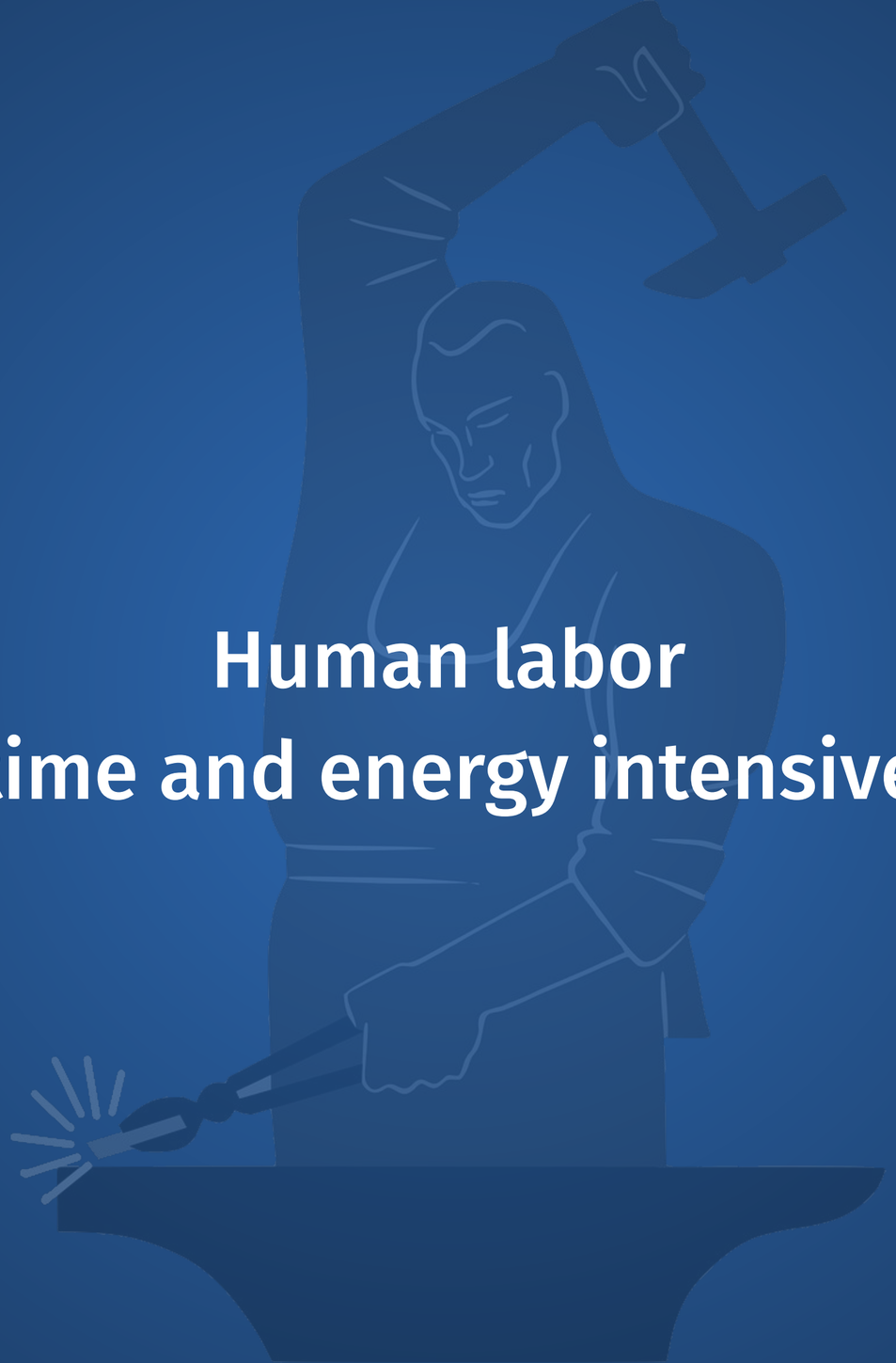



ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003



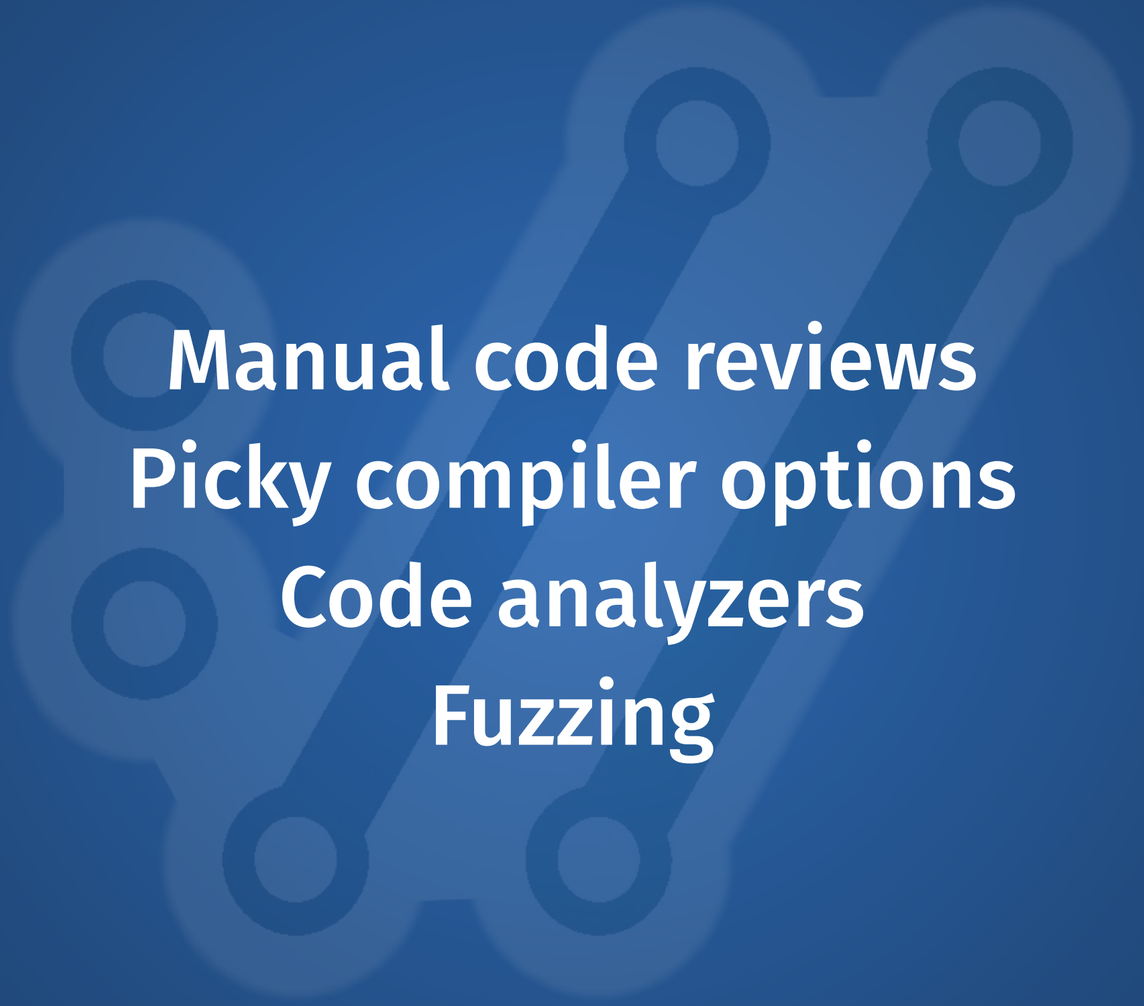
The artisanal era

Writing code



Human labor
time and energy intensive

Verifying code



Manual code reviews

Picky compiler options

Code analyzers

Fuzzing

Finding security flaws



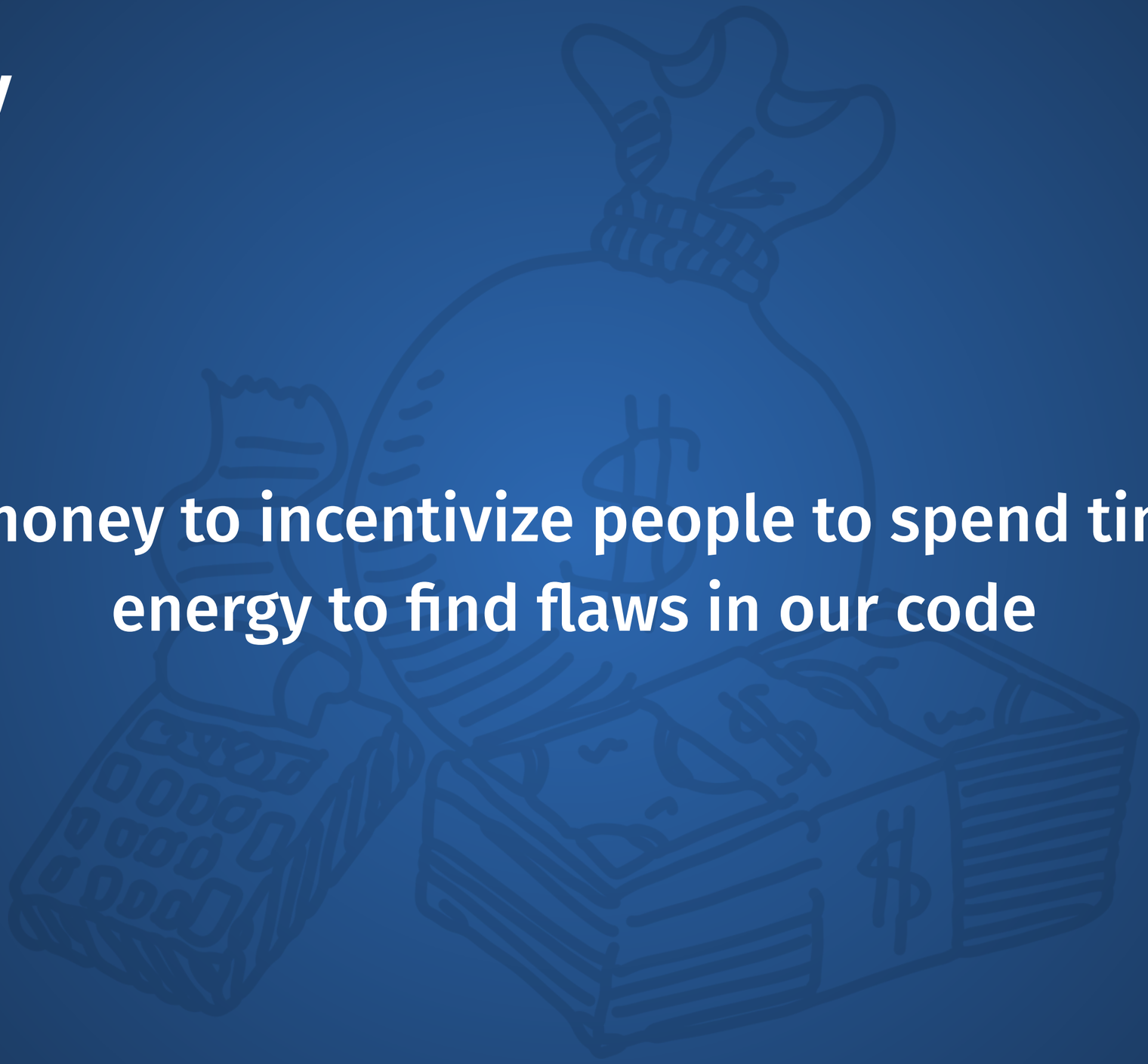
Human eyes

grepping

custom tools

time and energy intensive

Bug-bounty



Offer money to incentivize people to spend time and energy to find flaws in our code

Securing curl

Daniel Stenberg

Code style	Banned functions	Complexity checks	Human reviews	Review bots	No binary blobs	REUSE compliant
No git force push	No confusable Unicode	Document everything	Many tests	*torture tests*	CI like crazy	All picky compiler options and -Werror
Valgrind and sanitizers	AI + static code analyzers	Fuzzing, in CI and non-stop	read-only CI jobs	zizmor the CI jobs	Reproducible releases	Signed releases, commits, tags
git backup on codeberg	Vulnerabilities fixed in next release	Document vulnerabilities thoroughly	Code audits	(strong) 2fa for all committers	API and ABI stability allows *always-update*	private vulnerability reporting

Everything done in the open - accessible and transparently

The awakening

Generative AI

The Floodgates Open

Anatomy of a slop report

Confident tone

Overly polite

Zero substance

Hallucinated APIs

Impossible buffer overflows

Politeness covering for technical nonsense

An AI example

0x02 Proof-of-Concept Code

Inspect registers and stack:

Code 37 Bytes — Unwrap lines Copy Download

```
1 (gdb) info registers
2 (gdb) info frame
```

Signs of memory overwrite:

- r15 shows
- Recursive

0x04 Memor

Core Dump In

Code 57 Bytes — Copy Download

```
1 gdb curl core -q -ex "x/10i $rip - 0x10" -ex "info frame"
```

Analysis shows:

- Return address overwritten
- Stack recursion at `ngtcp2_http3_handle_priority_frame`

0x06 Risk Summary

- Affected Software: curl 8.13.0 (HTTP/3 enabled)
- Trigger: Stream dependency loop (e.g., stream 3 depends on 7, and 7 depends on 3)
- Result: Heap layout corruption, segmentation fault, denial-of-service
- Risk: High (pre-authentication, remote-triggerable)

```
28     asyncio.run(run_server())
```

The mentioned bad function **does not exist**

The gdb session is **made up**

The crash **does not happen**

The shown register contents are **made up**

Nothing in this report is **real**

LIES

Signal to noise bankruptcy

volume ↑

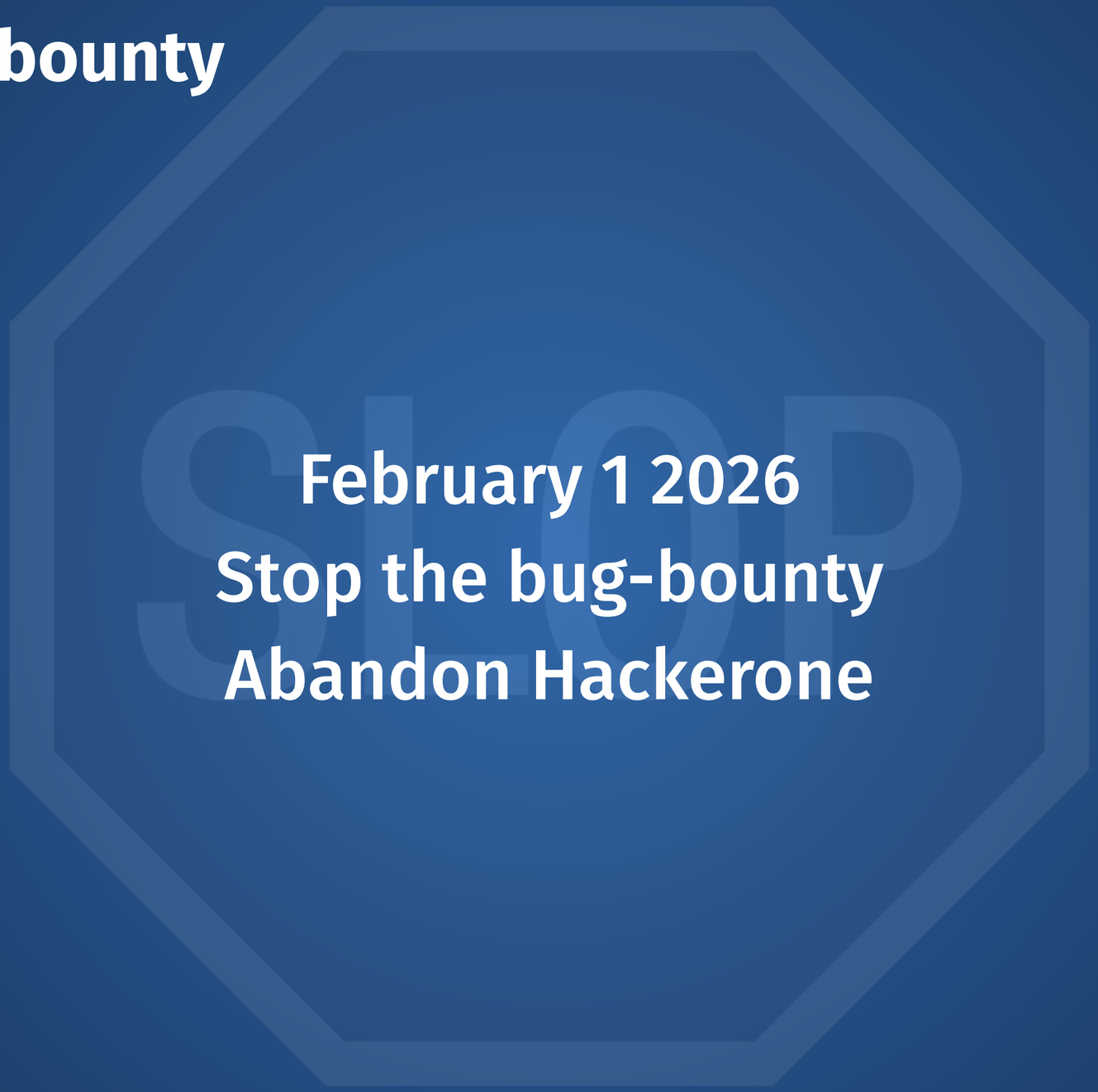
quality ↓

The mental taxation

Triaging "ridiculous levels" of reports.

The burnout of saying "This isn't real" many times a day.

Stop the bug-bounty



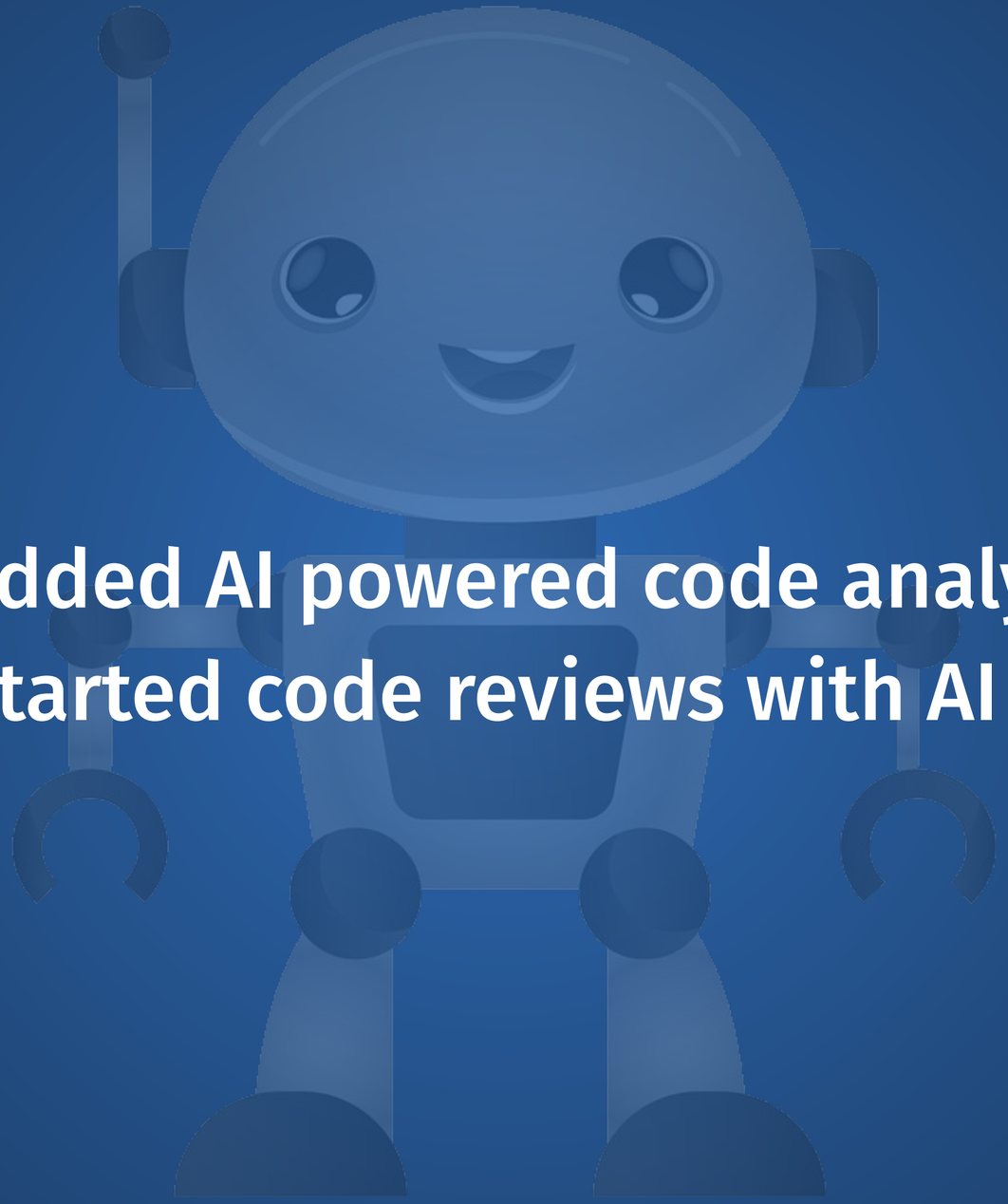
February 1 2026

Stop the bug-bounty

Abandon Hackerone

High-Quality Chaos

More AI tools



We added AI powered code analyzers

We started code reviews with AI bots

The reports are back

The rate of reports does not slow down

The quality looks fine !?

The slop rate stays low

Quality and volume

High quality in large volumes

Bug vs Vulnerability

AI is great at finding bugs.
Still weak at understanding security context.

Duplicates

More than one person run the same tools...

Technically correct

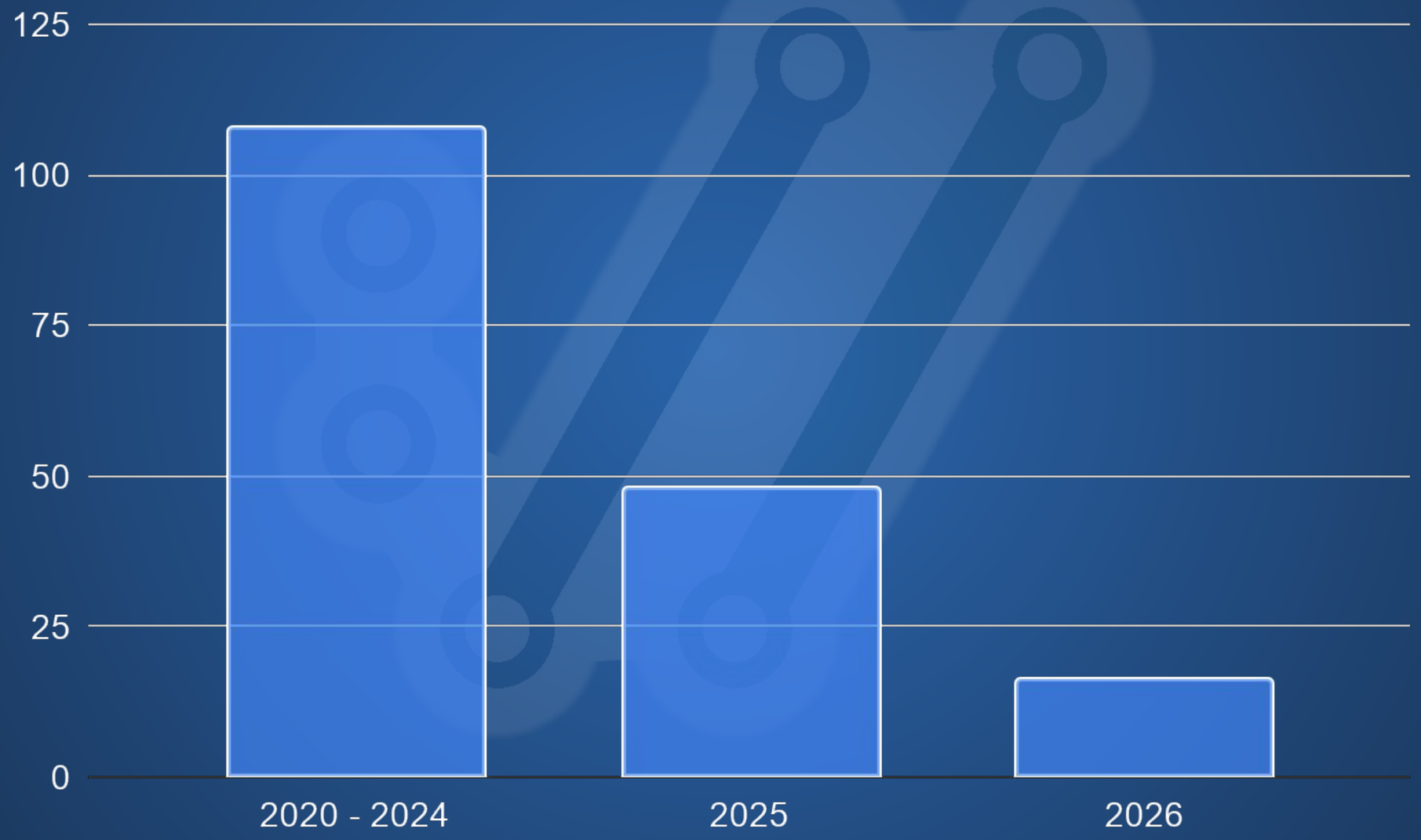
well-formatted and technically correct

Polished, professional, often wrong about security

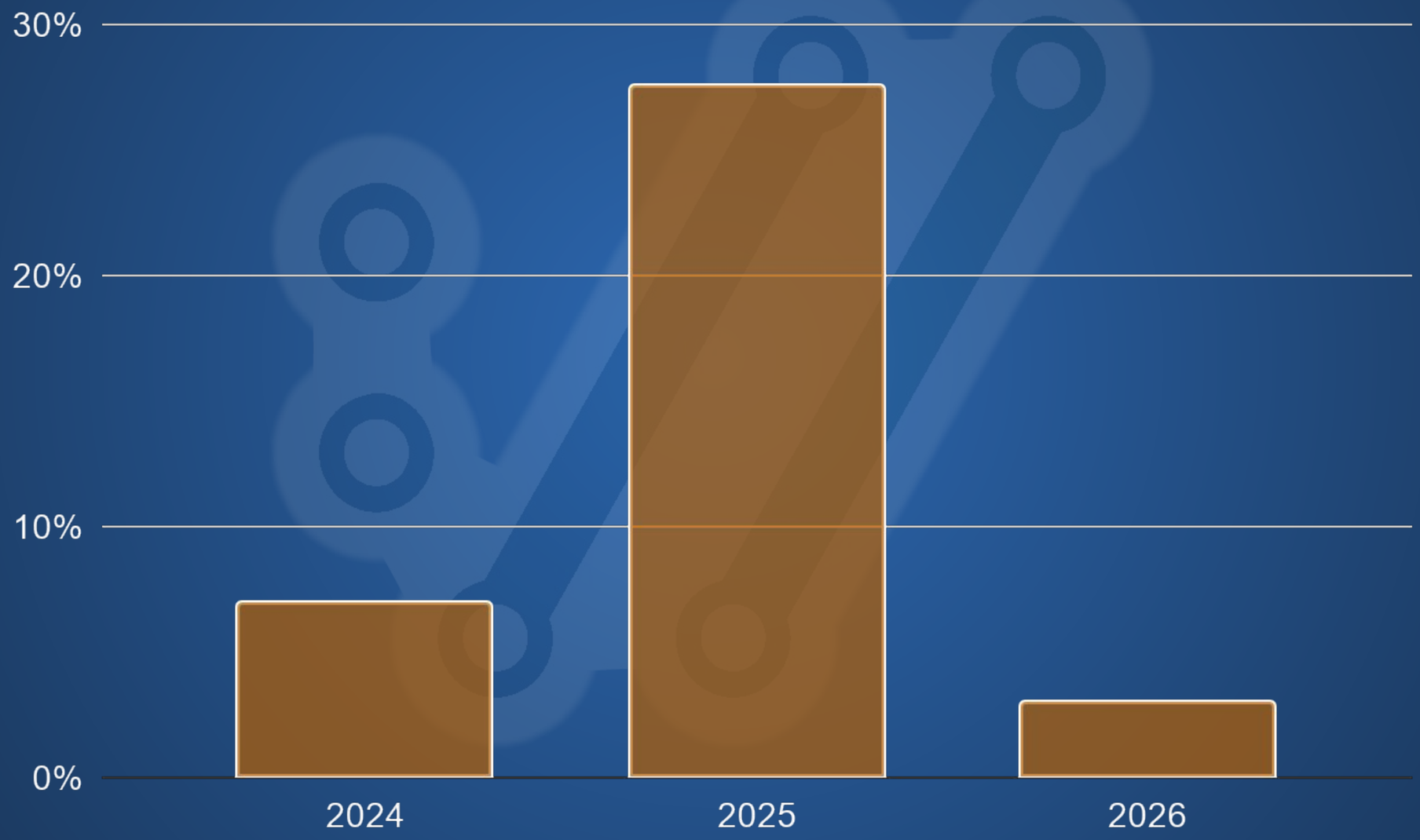
Accurate often enough

Numbers

Number of hours between vulnerability reports

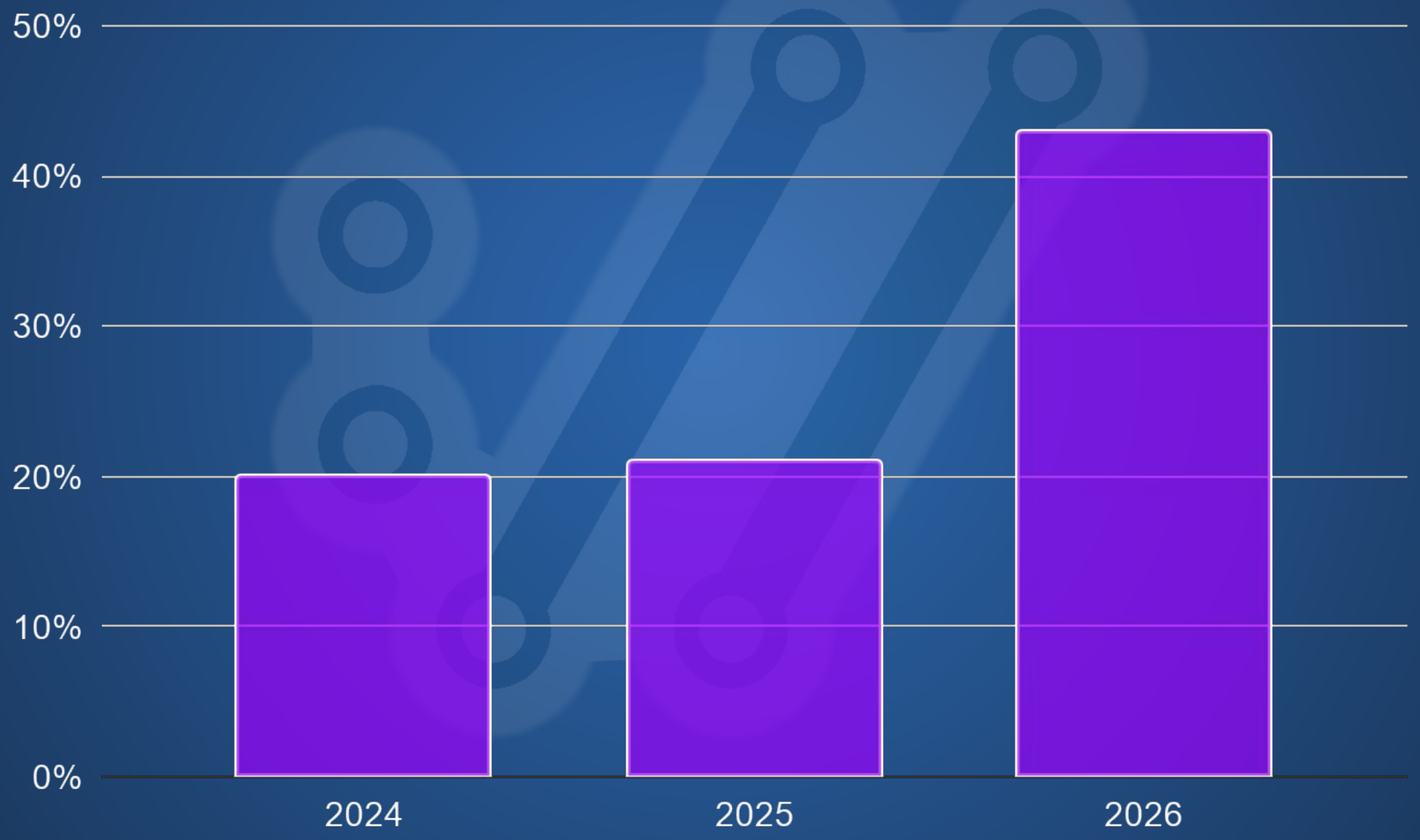


AI slop rate

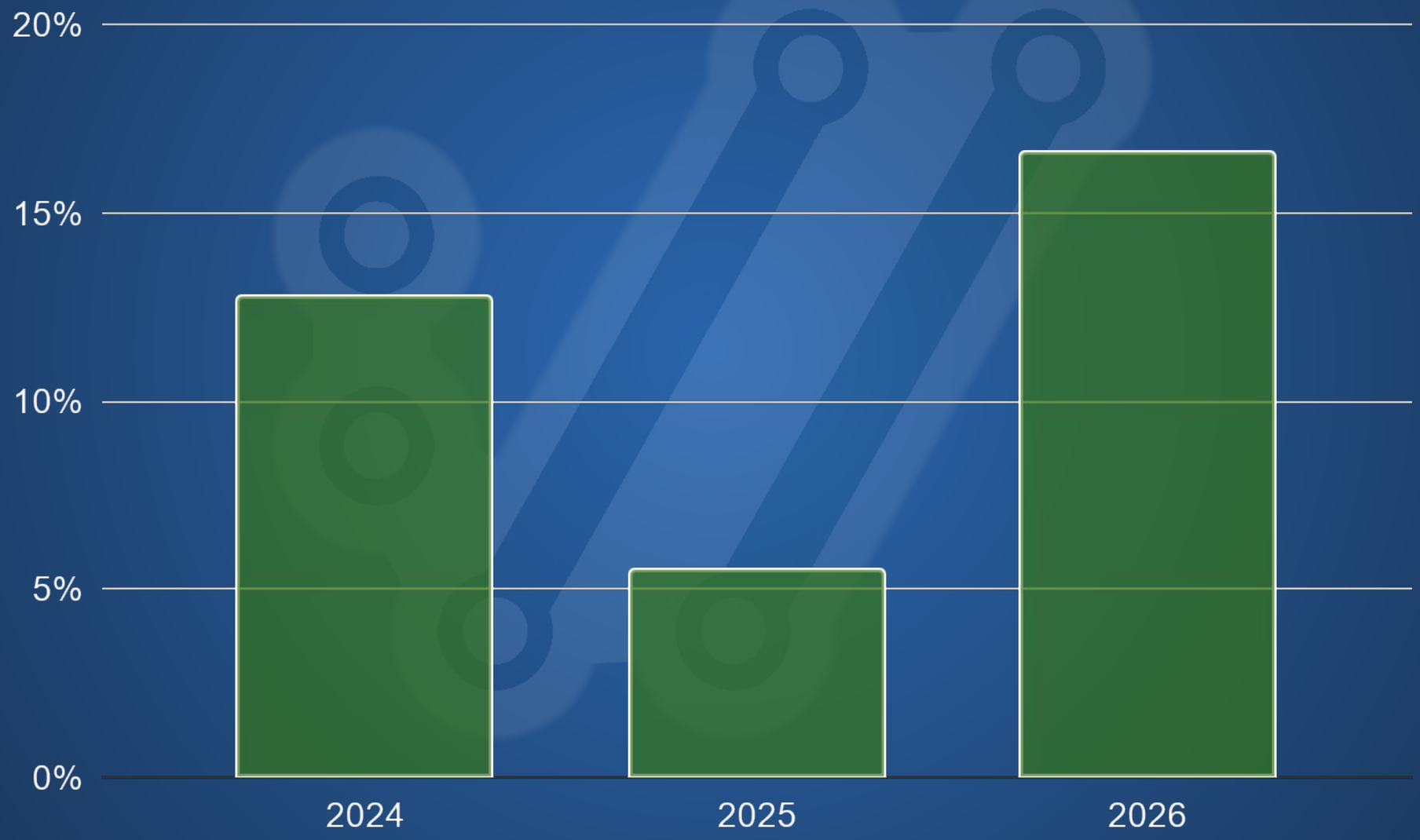


Daniel Stenberg

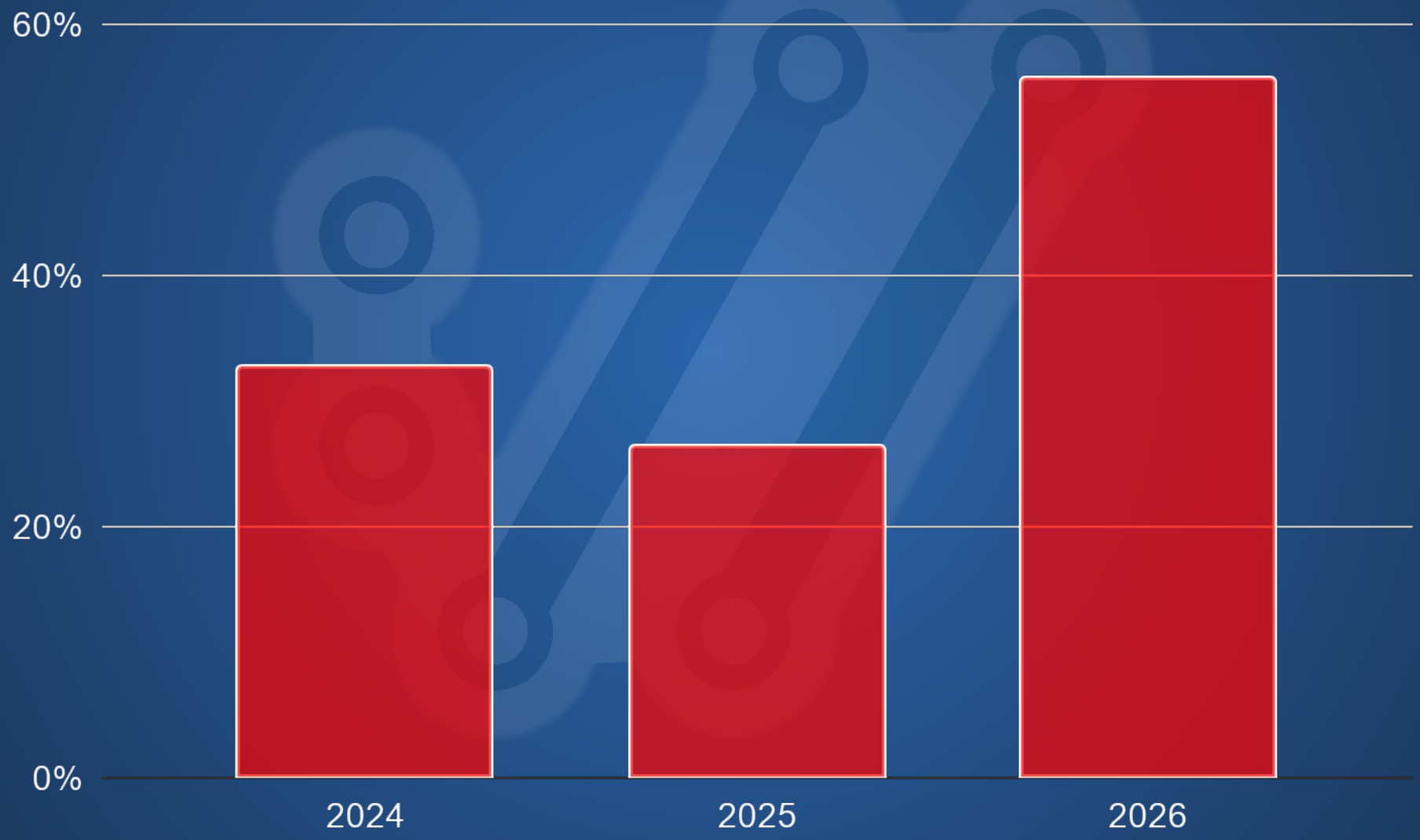
Share of reports that were bugs, not vulnerabilities



Confirmed vulnerability rate



Either vulnerability or a bug



Every report

Easy to generate

hard to process

rarely contain fixes

Every report

Assess, understand, is it a vulnerability?

Hours *to days* per issue

Higher quality reports take more time

Once confirmed, the fix itself is *usually* quick

Only curl vulnerability reports

not seen for regular issues

not seen for pull requests

Improvements compared to “before”

Not restricted to “has to build”

Not restricted to specific platforms

“Sees” into dependencies

“Knows” spoken protocols

Good at spotting broken patterns

Takes comments into account

Generates patches and test cases

Builds reproducers



The drowning

Everyone

Apache httpd, BIND, curl, Django, Elasticsearch Python client, Firefox, git, glibc, GnuTLS, GStreamer, Haproxy, Immich, libssh, libtiff, Linux kernel, OpenLDAP, PowerDNS, python, Prometheus, Ruby, Sequoia PGP, strongSwan, Temporal, Unbound, urllib3, Vikunja, Wireshark, wolfSSL, ...

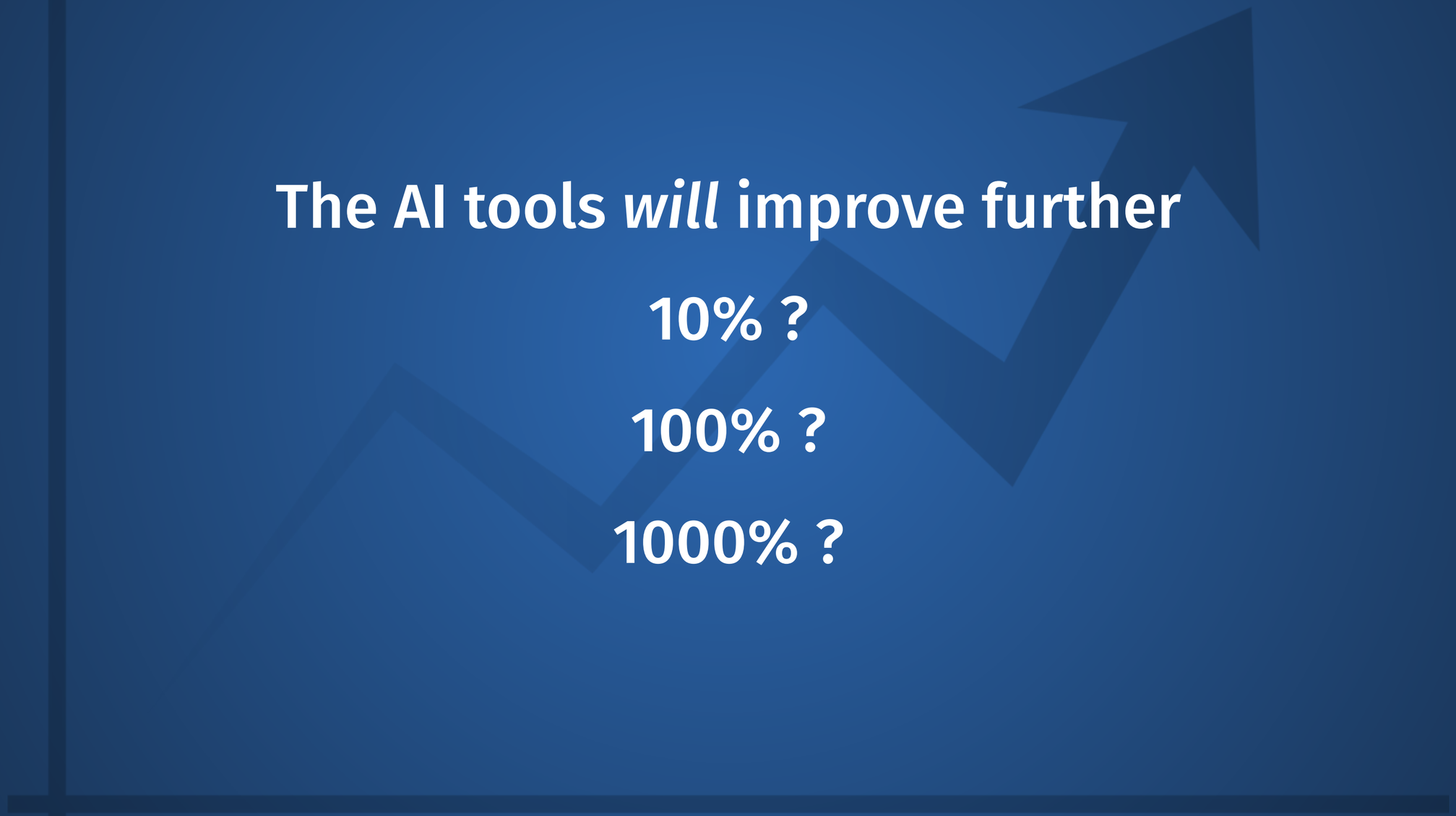
Security report overload



Understaffed

Underfunded

Overworked



The AI tools *will* improve further

10% ?

100% ?

1000% ?

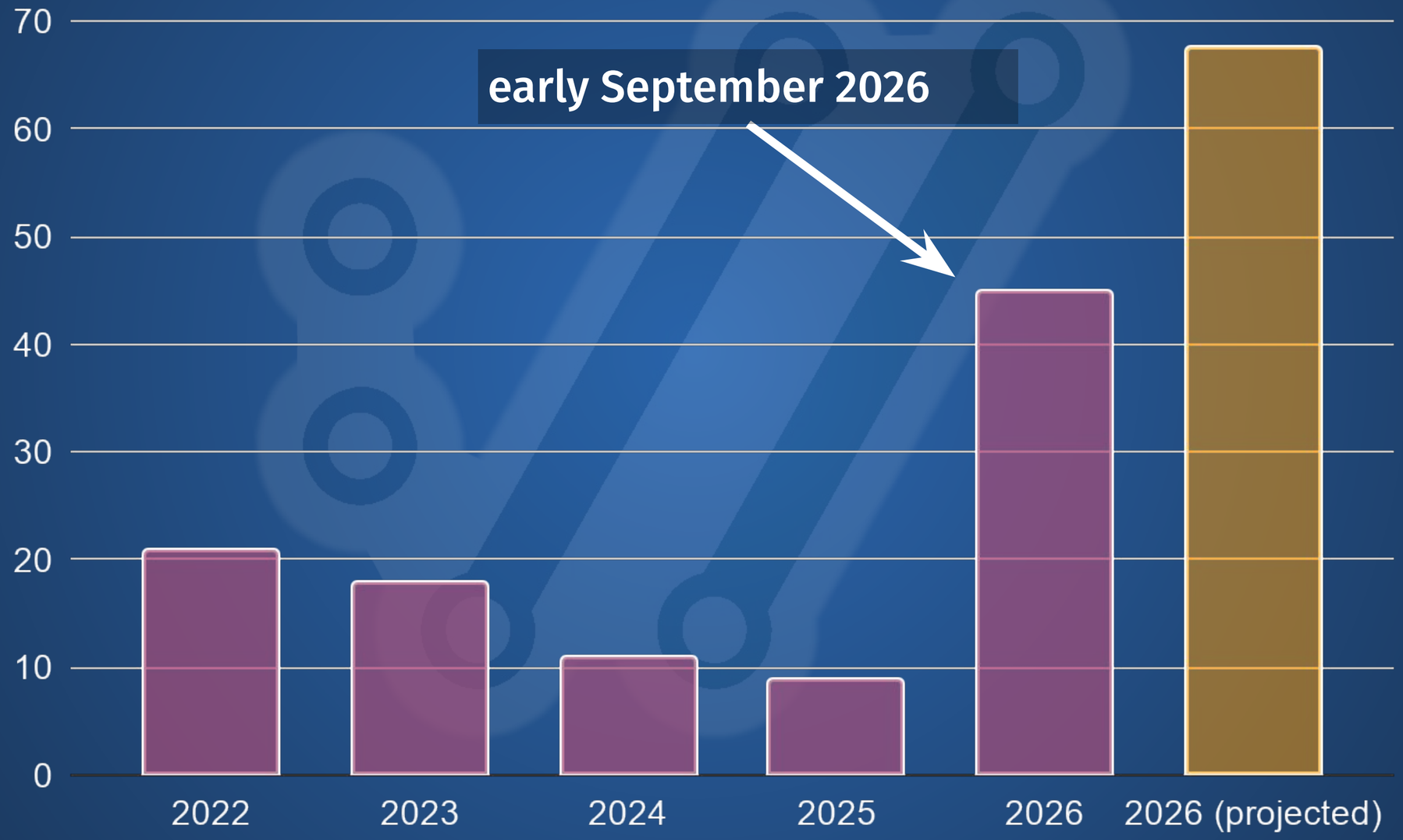
Daniel Stenberg



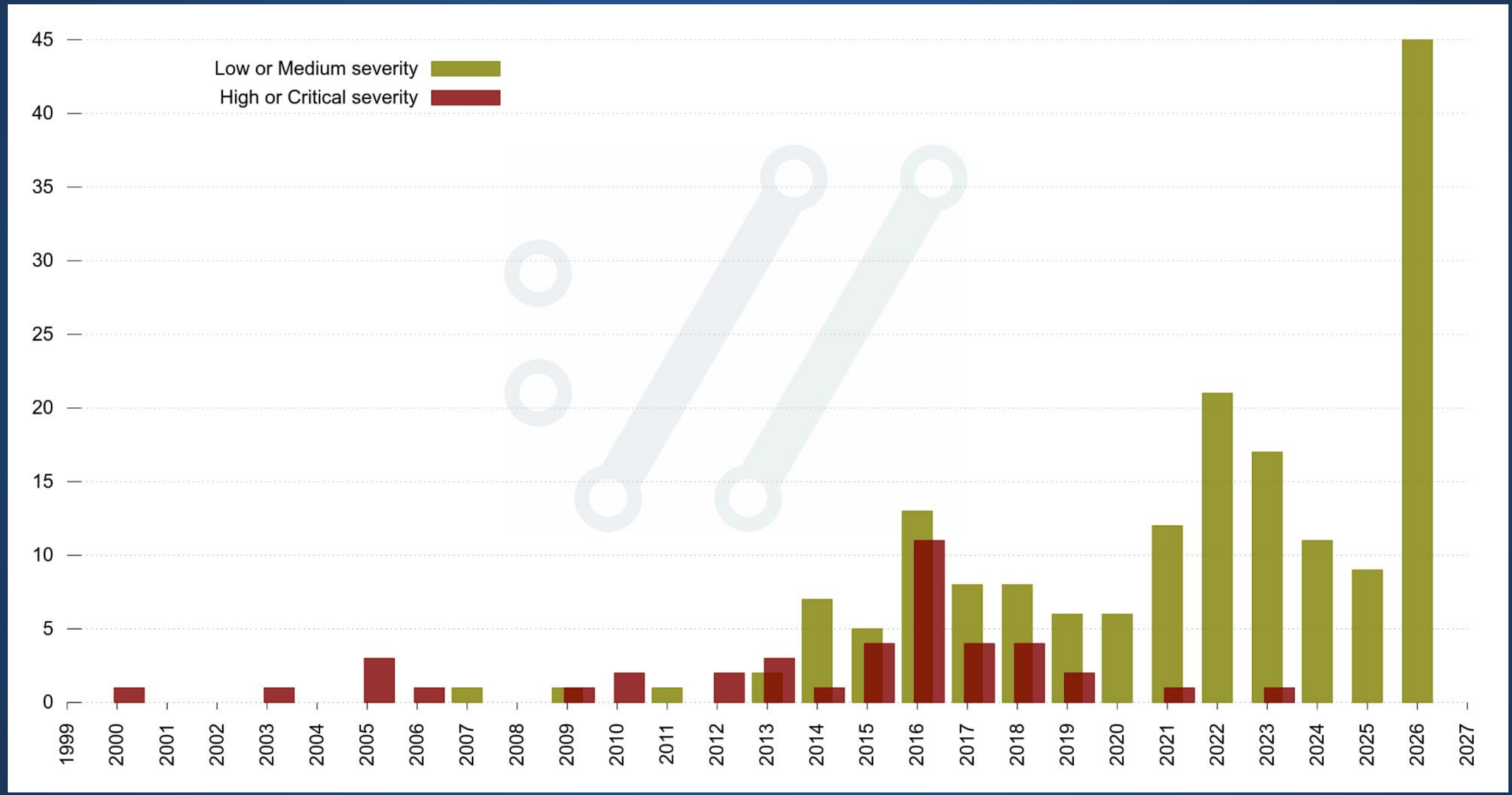


The report load will increase further

Number of published vulnerabilities



New findings tend to be less serious



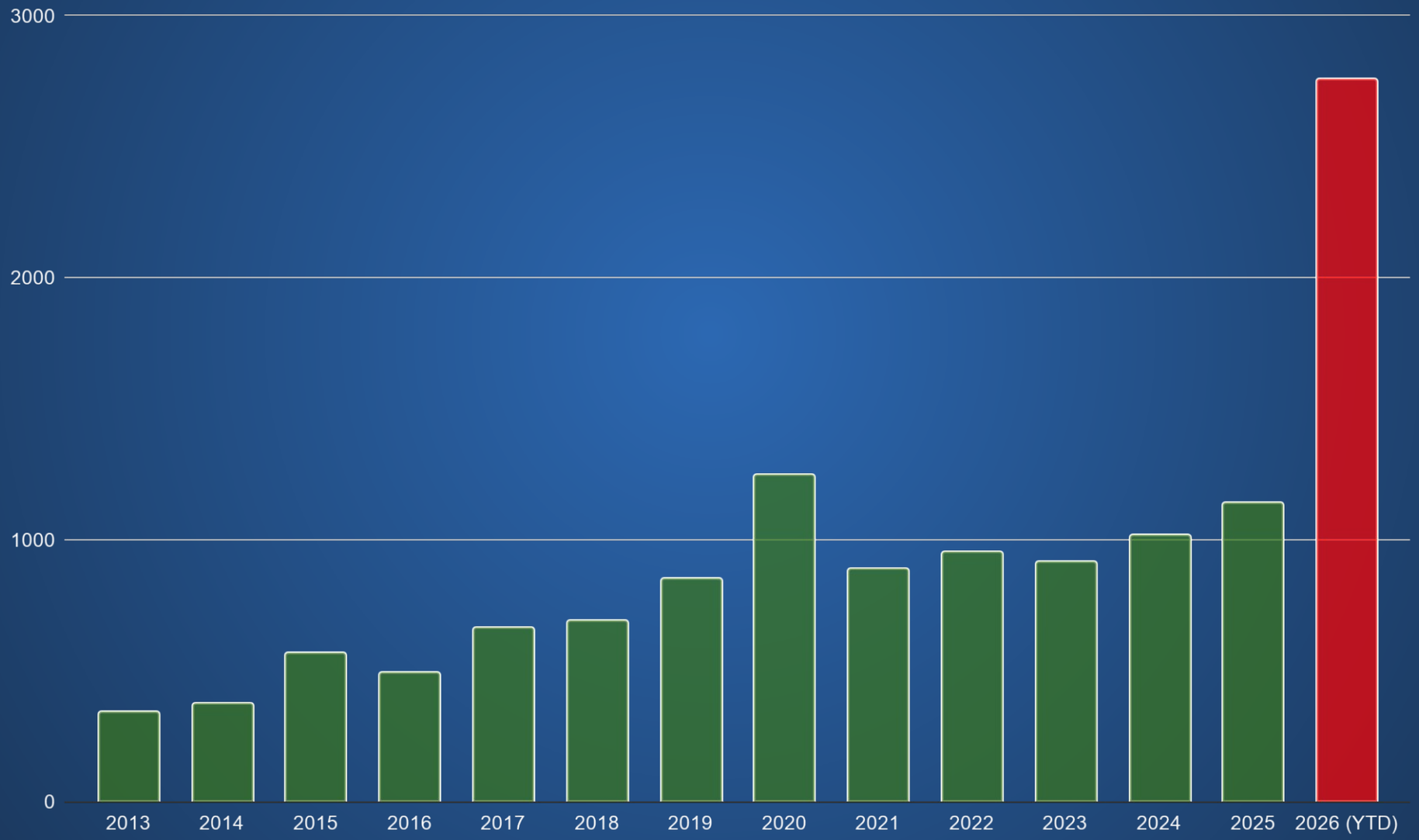
Firefox Security Bug Fixes by Month

All Sources ▪ All Severities



Daniel Stenberg

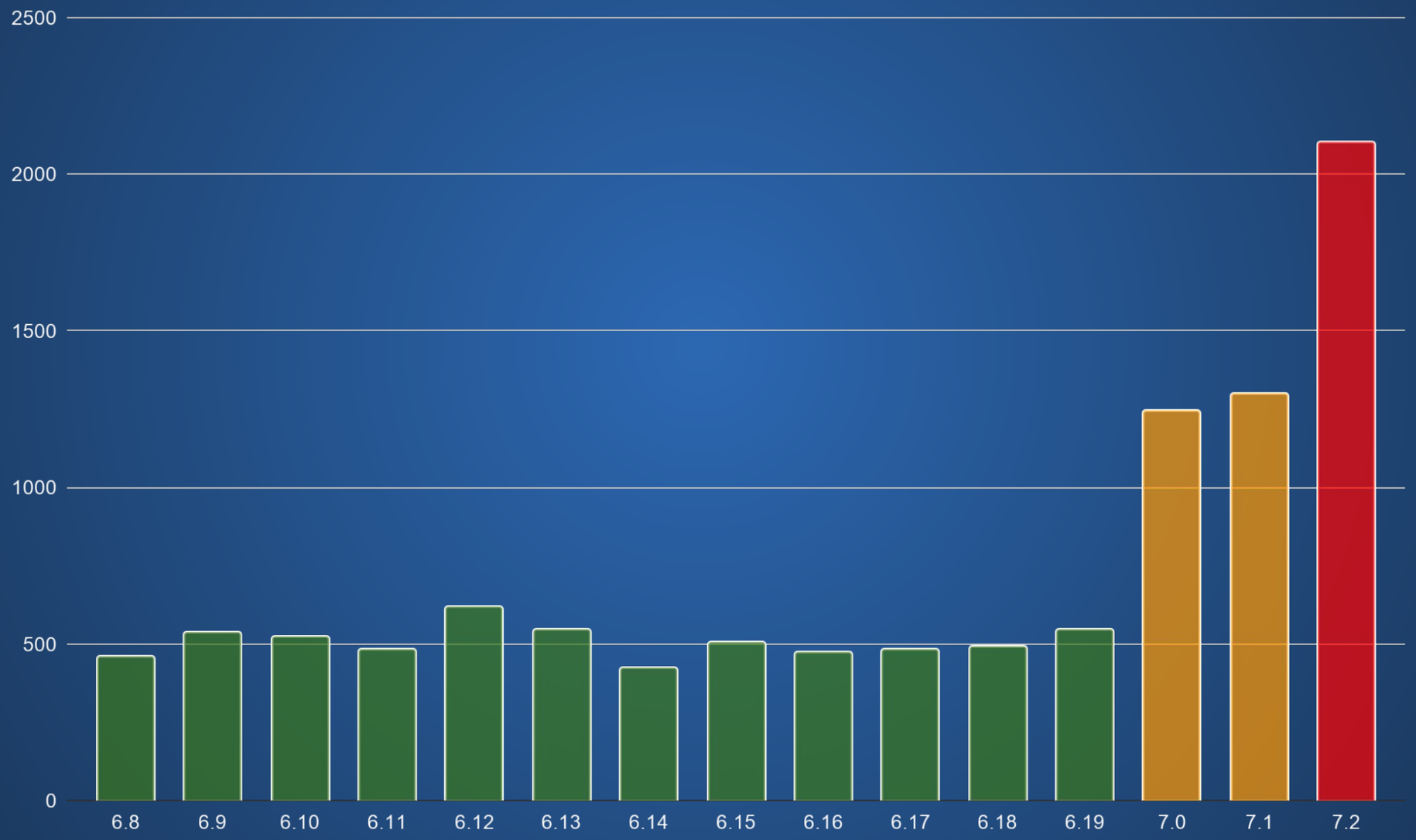
Microsoft vulnerabilities



Microsoft

Daniel Stenberg

Number of CVEs per Linux kernel release



Greg Kroah-Hartman

Spring 2026

Firefox fixed 271 vulnerabilities

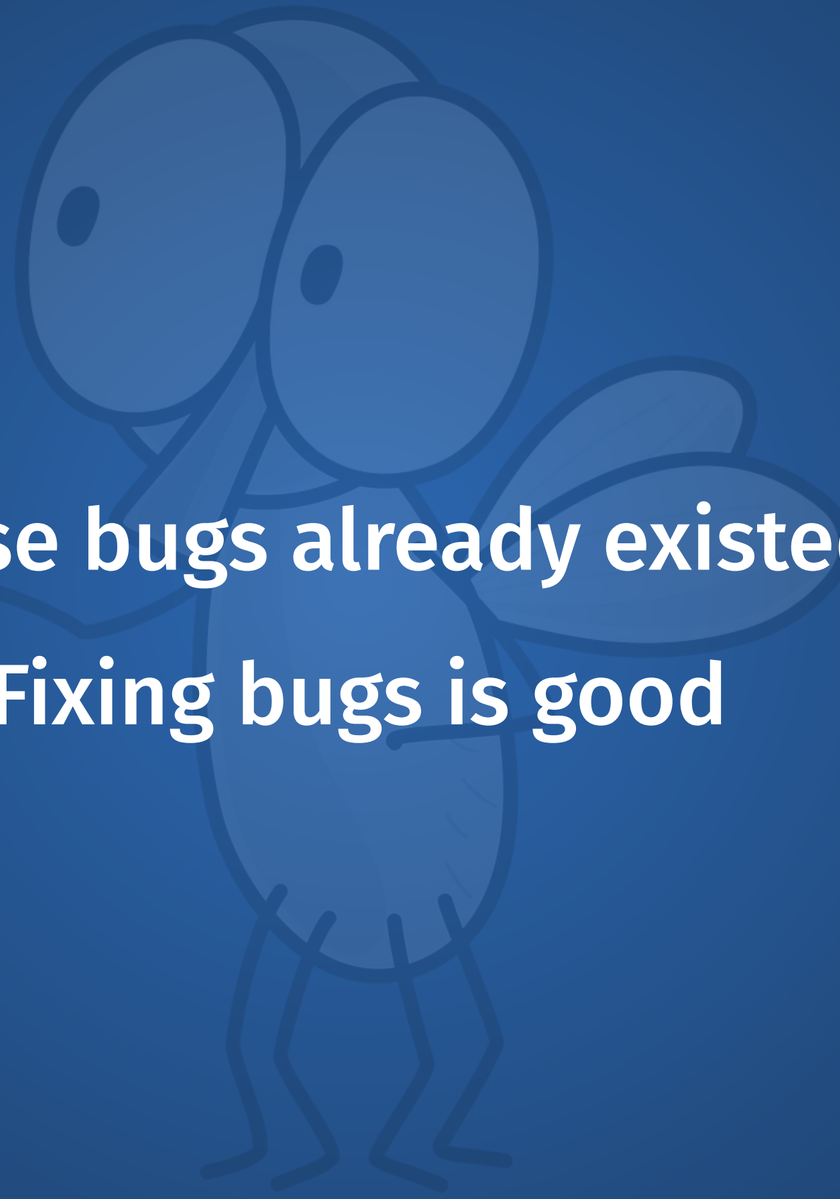
Microsoft fixed a near-record 167 security flaws

Apple addressed at least 52 vulnerabilities (up from 20)

Oracle addressed at least 450 flaws

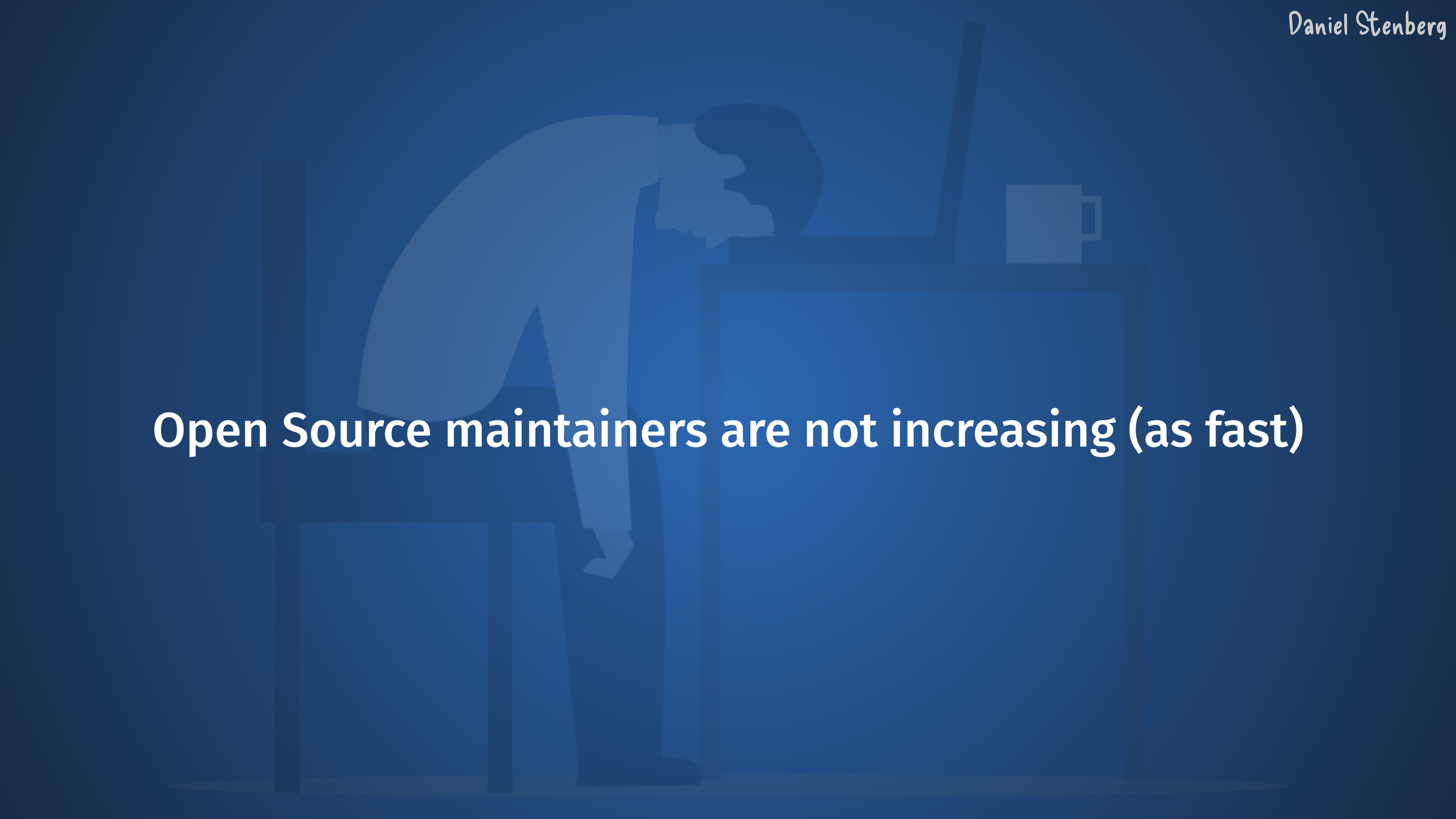
Chrome fixed 127 security flaws (up from 30)

Linux Kernel got *Copy Fail*, *Dirty Frag* and *Fragnesia*



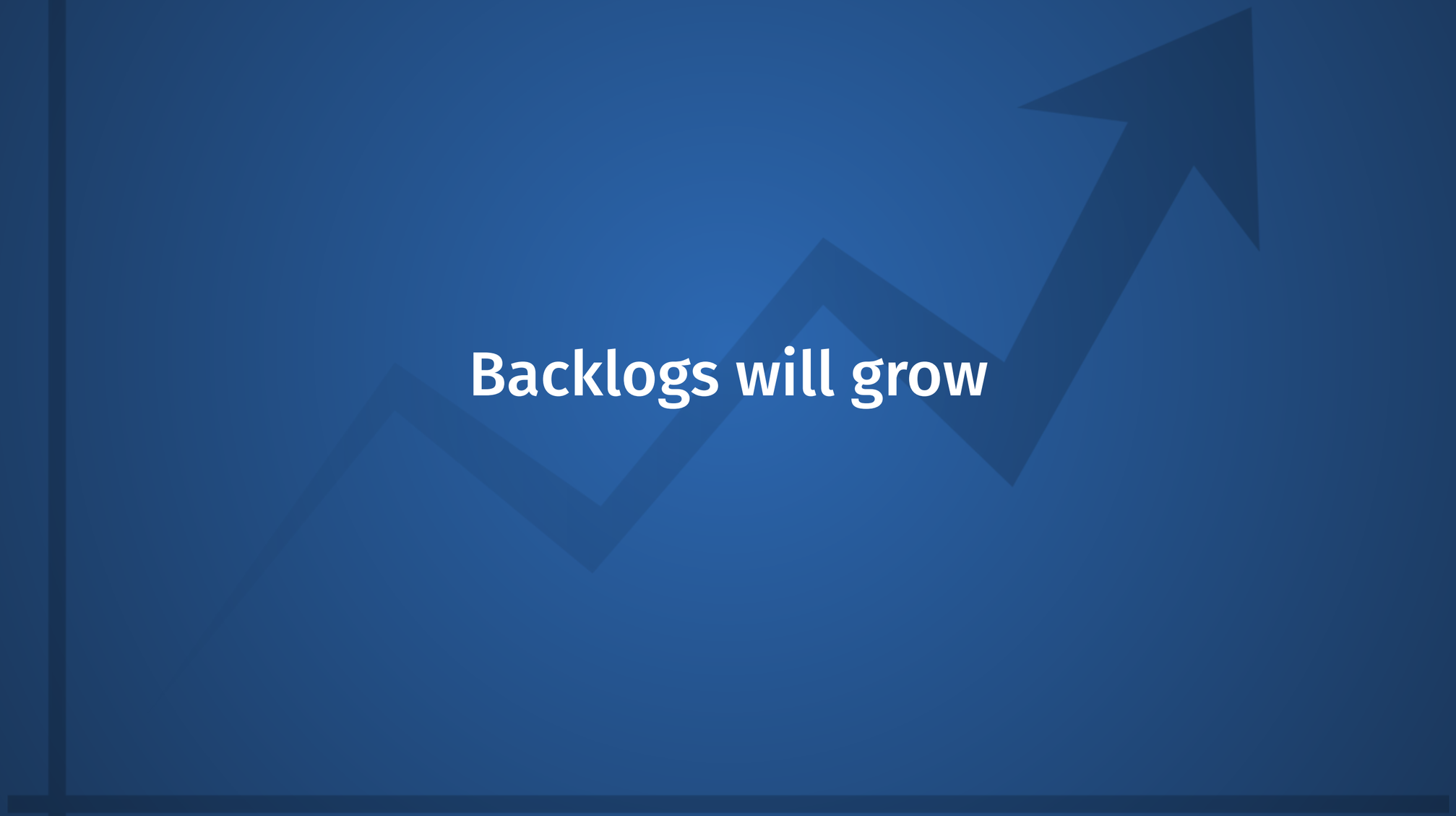
These bugs already existed

Fixing bugs is good



Open Source maintainers are not increasing (as fast)

Companies still likes free as in free beer



Backlogs will grow



more vulnerabilities will be published



Opportunity for malicious actors

"An AI scan of a 1.8-million-line enterprise application cost $315 in API fees but produced 3,560 findings, including 1,000 high-severity findings. Spending 30 minutes of security engineer triage per finding, the labor cost to review those results reached $128,000 before fixing a single issue"

Vulnerability-chaining



Faster release cycles?



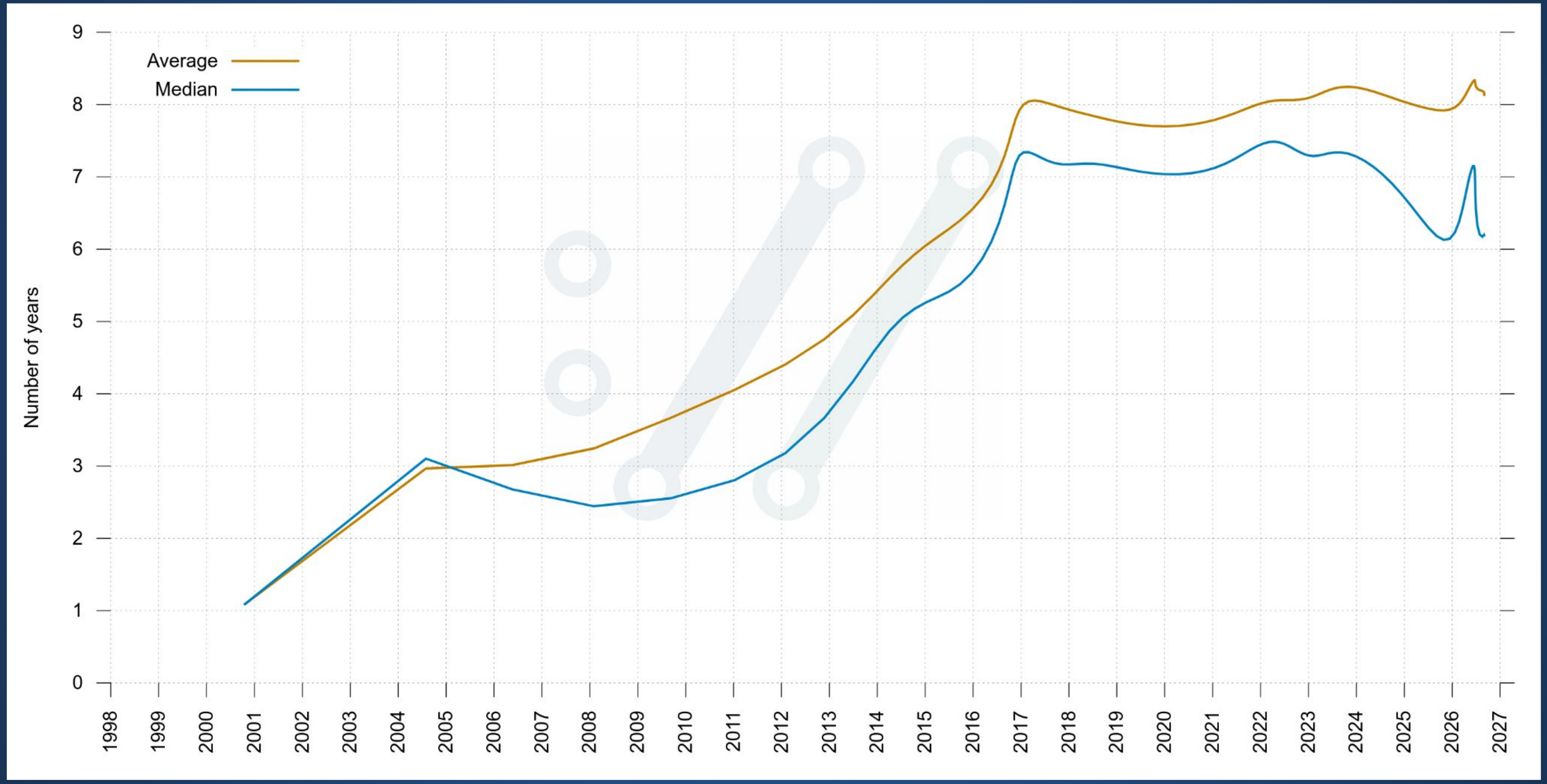
Going proprietary is no fix

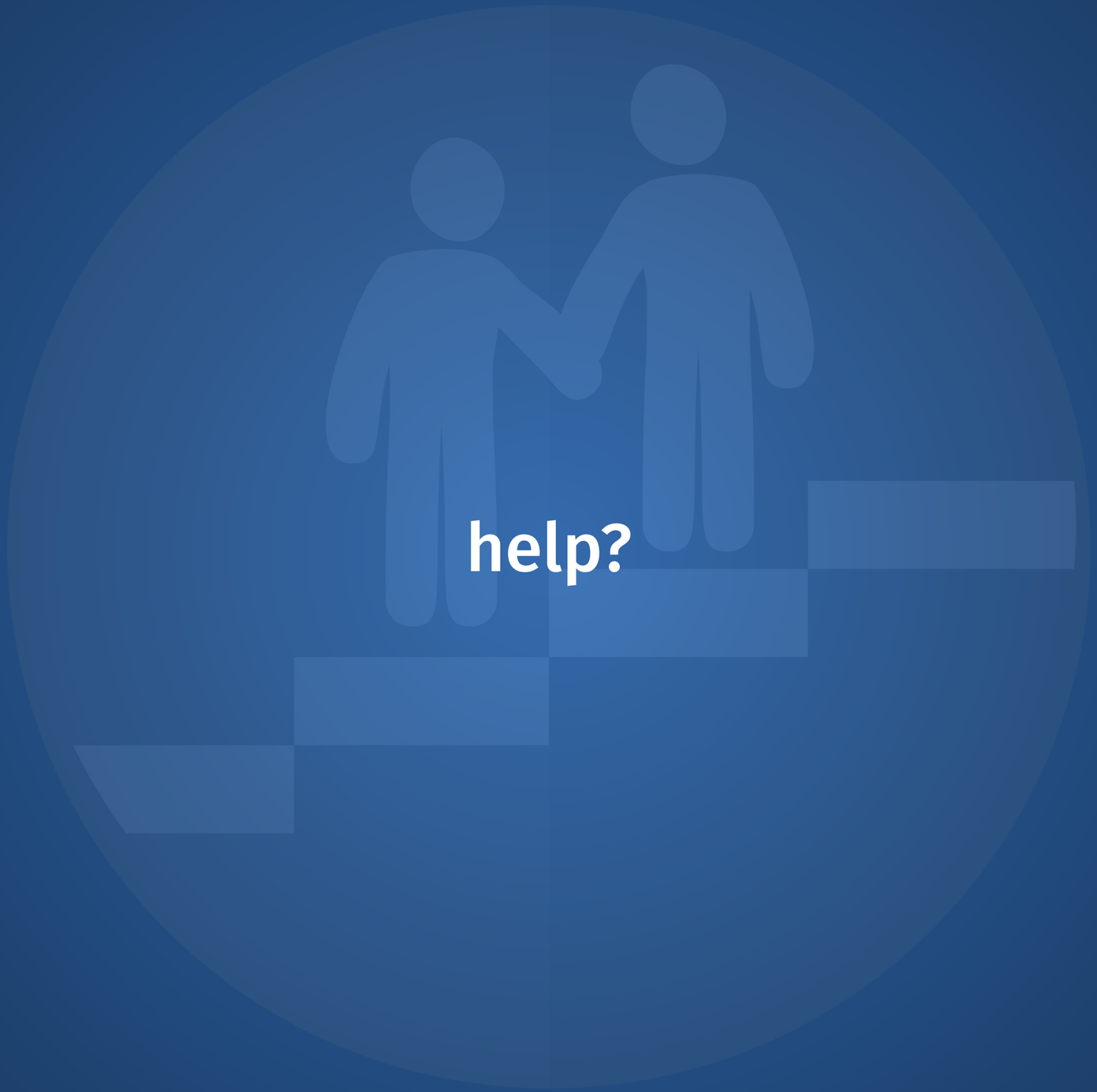


Can *you* patch fast enough?

Eventually the wave probably slows down

Vulnerability age when reported





help?


QUESTIONS?

License

This presentation and its contents are licensed under the Creative Commons Attribution 4.0 license:
http://creativecommons.org/licenses/by/4.0/