SharkFest'26 Europe

AI Found Holes In Our Cheese

Gerald Combs



Sponsored by:
PROFI TAP
DEEP NETWORK OBSERVABILITY
cPacket
TCP/IP training
endace
Record. Respond.

Expectation management

"Wireshark and AI" can and does mean lots of things.

This talk focuses on the interaction between one specific area of Wireshark development (vulnerability management) and one specific technology (AI vulnerability scanning).

This talk is not about packets. If you go to the session next door it won't hurt my feelings.

This talk is incomplete. Things are moving fast and there are lots of things; I can't speak authoritatively about all of them.

Many of you are more immersed in AI than I am. Feel free to speak up.

Wireshark 4.6.5 fixed 42 vulnerabilities

How was *your* spring?

Is 42 a lot?

Compared to a normal Wireshark release? Yes! by an order of magnitude.

Compared to other software? I honestly don't know.

We have 7 million lines of code. 42 / (7000000 / 1000) = 0.006 defects per kLoC.

We've had 101k commits and issued 672 advisories since our inception (July 1998).

101616 / 672 = 1 advisory every ~151 commits.

...so we should just delete every 151st commit everything will be perfectly fine, right?

This is part of a larger trend

FIRST Mid-Year Vulnerability Forecast Confirms Historic Surge, Projects ~66,000 CVEs in 2026

AI and a rapidly expanding CVE ecosystem blow past February's already-record projections by 46%, but exploitable risk remains flat

DENVER, June 15, 2026 – Today at FIRST's 38th Annual Conference, the Forum of Incident Response and Security Teams (FIRST) released its 2026 Mid-Year Vulnerability Forecast, revealing that actual CVE disclosures are already running 46.3% above the projections published just four months ago.

https://www.first.org/newsroom/releases/20260615

Year	2026	2025	2024	2023	2022	2021	2020	2019	2018
Count	105,782	70,729	52,316	40,051	34,553	28,506	30,680	24,179	21,407

https://www.cve.org/about/Metrics

What is Wireshark?

The world's most popular network protocol analyzer.

Used by network engineers, security teams, developers, and educators.

At its core, a very powerful data dissection engine.

3400 protocols

7 million lines of code

What is Wireshark?

The world's most popular network protocol analyzer.

Used by network engineers, security teams, developers, and educators.

At its core, a very powerful data dissection engine.

3400 ~~protocols~~ways to be "clever" about serialization

7 million lines of ~~code~~attack surface

Attack surface

Classic C headaches like buffer overflows, NPEs, and uninitialized memory.

Sometimes protocols are just difficult to handle safely.

“Difficult” includes things like recursion. ASN.1 sequences, DOCSIS, JPEG IFDs, many others.

Computer numbers are either uselessly small or unreasonably large.

Loops don’t have to be infinite to cause trouble. What the heck are you supposed to do with a 32-bit loop counter when most of the high bits are set?

How We Secure Wireshark

API safety (tvbuffs, wmem)

Automatic code generation

Code review

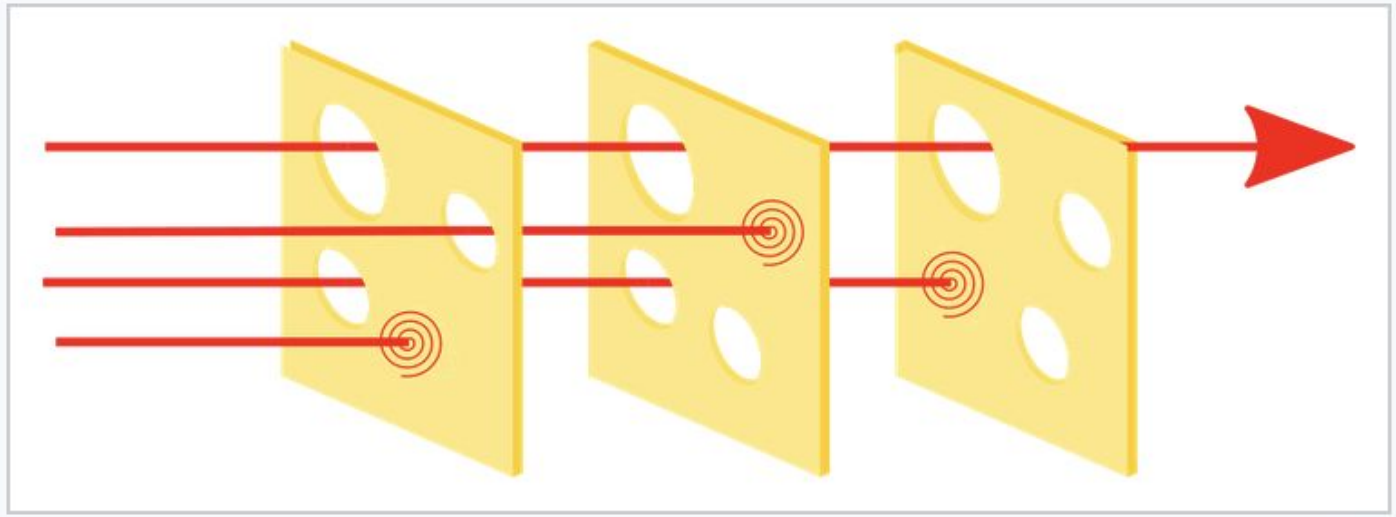
Continuous integration tests

Fuzzing (us, OSS-Fuzz, others)

Static analysis (Black Duck's Coverity scan, clang, gcc, our own, others)

Researchers aka the kindness of strangers

AI found holes in our cheese



The Swiss cheese model of accident causation illustrates that, although many layers of defense lie between hazards and accidents, there are flaws in each layer that, if aligned, can allow the accident to occur. In this diagram, three hazard vectors are stopped by the defenses, but one passes through where the "holes" are lined up.

https://en.wikipedia.org/wiki/Swiss_cheese_model

“We should start looking for problems in our code”

1999 randpkt – Generate a .pcap with random blobs.

2005 editcap – Add errors to an existing .pcap.

2013 AFL – Guided fuzzing.

2015 OSS-Fuzz – Guided fuzzing at scale.

What is a vulnerability?

Ideally we'd be able to pick a CVSS[1] threshold.

However, CVSS is like olympic judging. Your score might not be my score.

Relatedly, CVEs and CVSS suffer from Goodhart's Law.

> There's a whole economy around "make number go up"

In our case, this has been a pretty good guideline:

> *Does this allow an attacker to ruin someone's day with little to no action on their part?*



1. Common Vulnerability Scoring System

The vulnerability discovery and reporting process

1. Someone (maybe us, maybe an outside entity) finds a flaw.
2. We fix the flaw in our master branch and backport it to supported release branches.
3. We create an advisory and assign it an identifier (wnpa-sec-YYYY-NN).
4. We request a CVE ID, which has its own number (CVE-YYYY-NNNNN).
5. The Wireshark advisory and CVE ID will be listed in the release notes.

This ~~happens~~happened about 0 to 5 times per release, and releases happen about every 6 weeks.

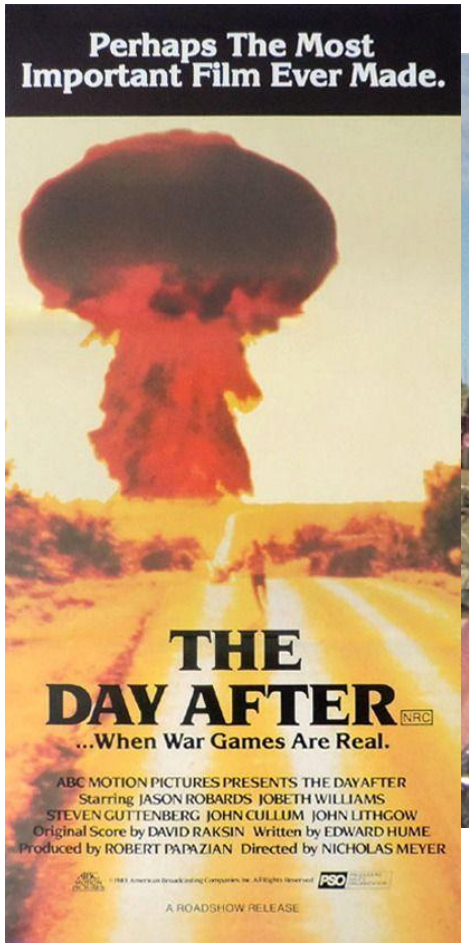
This work is important but tedious.

For a more formal version, see https://gcve.eu/bcp/gcve-bcp-02/

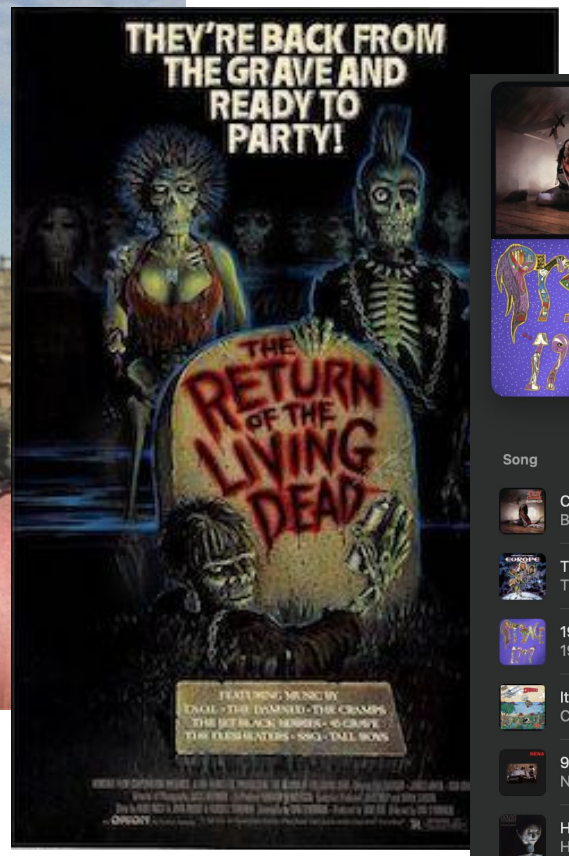
Also see SECURITY.md in the source repository.

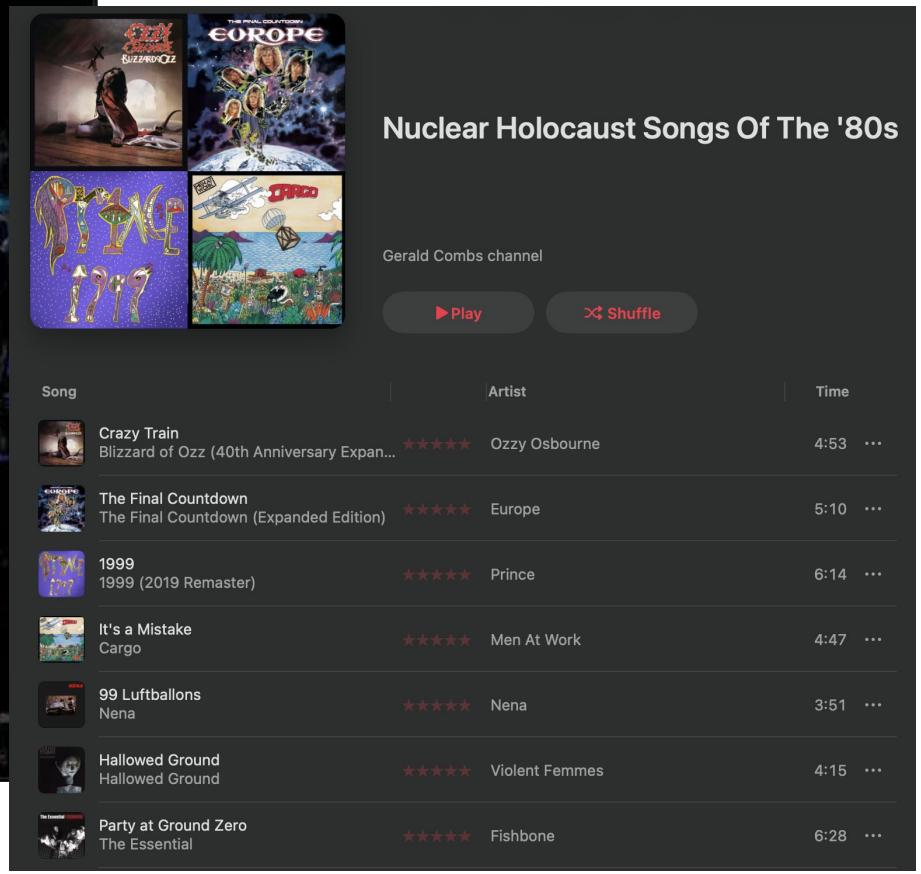
What if we get a whole bunch at once?

The apocalypse I was promised









The apocalypse I got

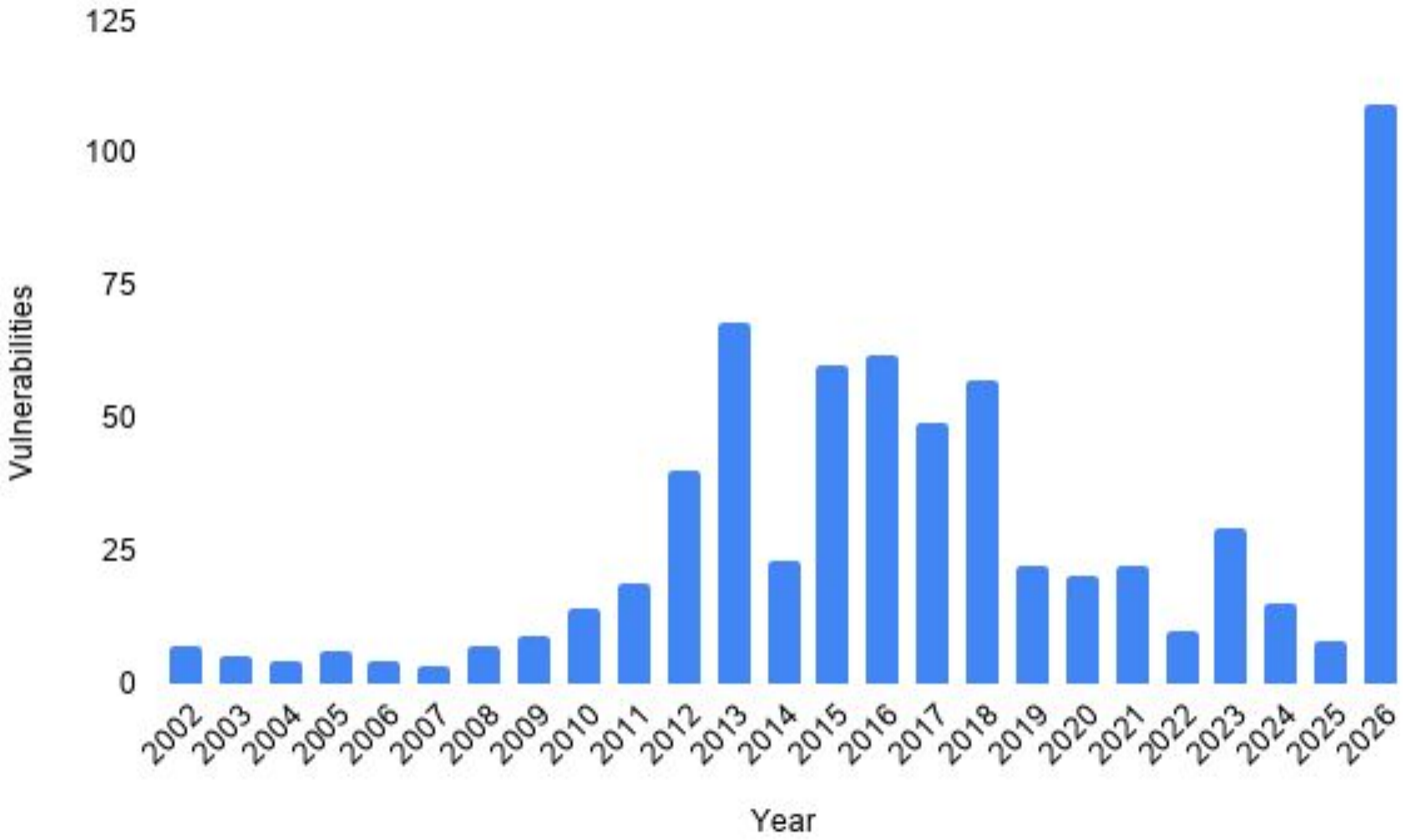
What's New

Bug Fixes

This release fixes quite a few vulnerabilities. This is due to to a recent trend in AI-assisted vulnerability reports.

- wnpa-sec-2026-08 Monero dissector crash. Issue 21066. CVE-2026-5409.
- wnpa-sec-2026-09 BT-DHT dissector crash. Issue 21067. CVE-2026-5408.
- wnpa-sec-2026-10 FC-SWILS dissector crash. Issue 21070. CVE-2026-5406.
- wnpa-sec-2026-11 SMB2 dissector infinite loop. Issue 21073. CVE-2026-5407.
- wnpa-sec-2026-12 ICMPv6 dissector crash. Issue 21077. CVE-2026-5299.
- wnpa-sec-2026-13 AFP dissector crash. Issue 21088. CVE-2026-5401.
- wnpa-sec-2026-14 TLS dissector crash and possible code execution. Issue 21090. CVE-2026-5402.
- wnpa-sec-2026-15 K12 RF5 file parser crash. Issue 21094. CVE-2026-5404.
- wnpa-sec-2026-16 SBC codec crash and possible code execution. Issue 21103. CVE-2026-5403.
- wnpa-sec-2026-17 RDP dissector crash and possible code execution. Issue 21105. CVE-2026-5405.
- wnpa-sec-2026-18 AMR-NB codec crash. Issue 21111. CVE-2026-5654.
- wnpa-sec-2026-19 SDP dissector crash. Issue 2111. CVE-2026-5655.
- wnpa-sec-2026-20 iLBC audio codec crash. Issue 21113. CVE-2026-5657.
- wnpa-sec-2026-21 Profile import crash and possible code execution. Issue 21115. CVE-2026-5656.

…and this wasn't even the first



What was different this time?

Until this spring, AI was terrible at finding security flaws. It got better.

AI found gaps in our fuzzing coverage.

> We focused on feeding fuzzed captures to the dissection engine, wood chipper style.
>
> It's difficult to fuzz some parts of the code.

Many issues in a gray area beyond "ruin someone's day".

We weren't the only ones.

Wireshark isn't just Wireshark

bcg729	libilbc	Lua	Qt
Brotli	libmaxminddb	minizip-ng	SBC
c-ares	libpcap	Nettle	Snappy
Dirent	libscap	nghttp2	SpanDSP
Gettext	libsinsp	nghttp3	Sparkle
GLib	libsmi	npcap	SpeexDSP
GnuTLS	libssh	Opus	WinSparkle
GMP	Libtasn1	OpenCORE AMR	zlib-ng
Libgcrypt	libxml	p11-kit	zstd
libgpg-error	Lz4	PCRE2	

Why I’m jealous of Rust and Go developers

Most modern programming languages have nice, featureful package managers.

C and C++ leave this as an exercise for the reader.

We get to wrangle a pile of dependencies on Windows and macOS in addition to Wireshark.

“The future is already here — it's just not very evenly distributed.”

William Gibson

Past performance does not guarantee future performance…

1999 randpkt – Generate a .pcap with random blobs.

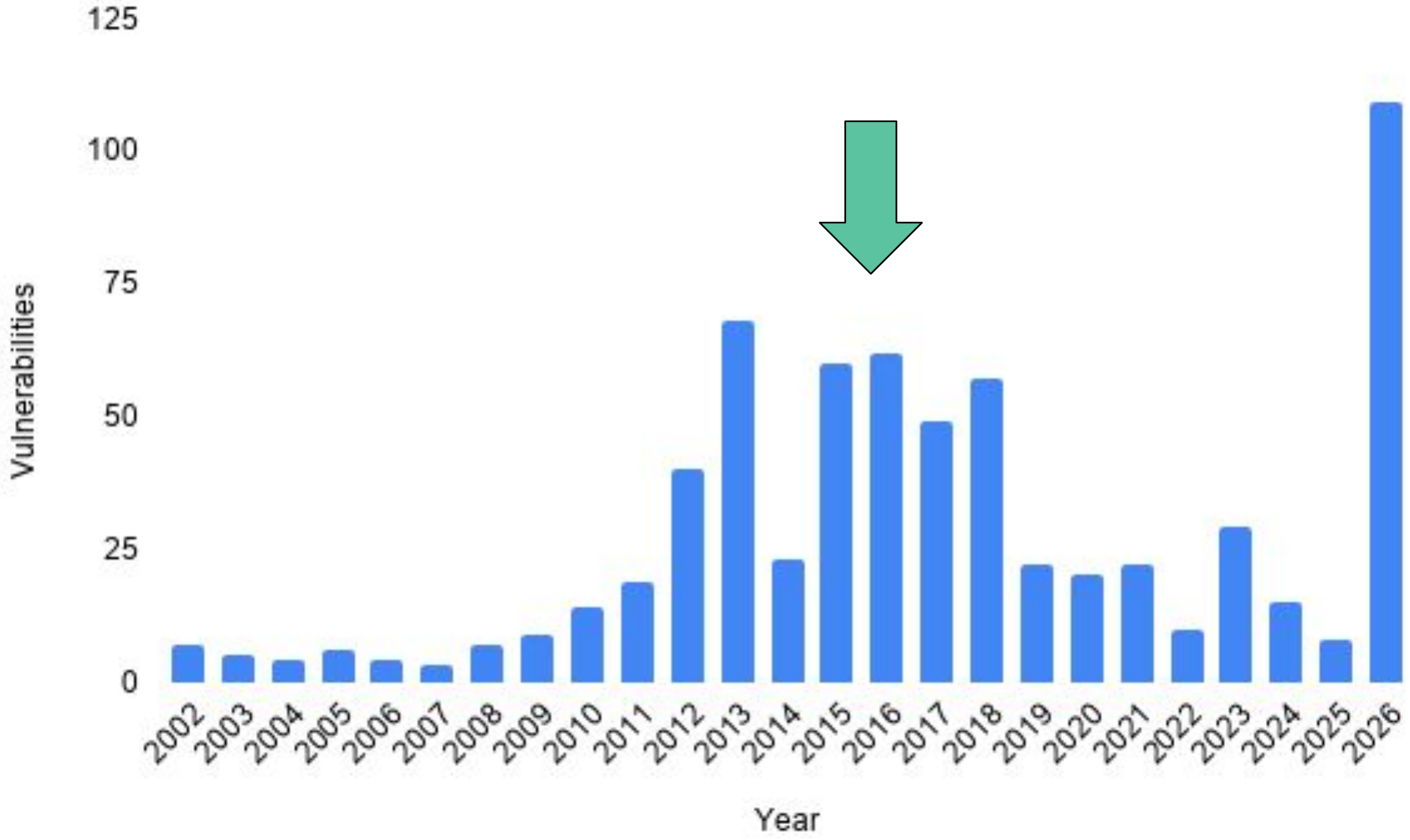
2005 editcap – Add errors to an existing .pcap.

2013 AFL – Guided fuzzing.

2015 OSS-Fuzz – Guided fuzzing at scale.



…however…


125
100
75
50
25
0
Vulnerabilities
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
Year

It’s not just Wireshark



https://cve.icu/years.html?years=2026

And it’s not just open source



Everyone is getting tired and sloppy

CVE-2026-98432

Publication date	15 October 2026
Last updated	29 October 2026

UBUNTU PRIORITY

High

Why this priority?

CVSS 3 SEVERITY SCORE

9.8 · Critical

Score breakdown

Description

In the Linux kernel, the following vulnerability has been resolved:
ipv4: free inet_opt and ireq_opt after an RCU grace period
tcp_v4_syn_recv_sock() transfers ownership of ireq->ireq_opt to the child socket (newinet->inet_opt) without copying it.
Another cpu can concurrently process a retransmitted SYN for the same request socket, and send a SYNACK from tcp_check_req(). tcp_v4_send_synack() and inet_csk_route_req() read ireq->ireq_opt under rcu_read_lock() only, and ip_build_and_send_pkt() and ip_options_build() then read opt->optlen twice.

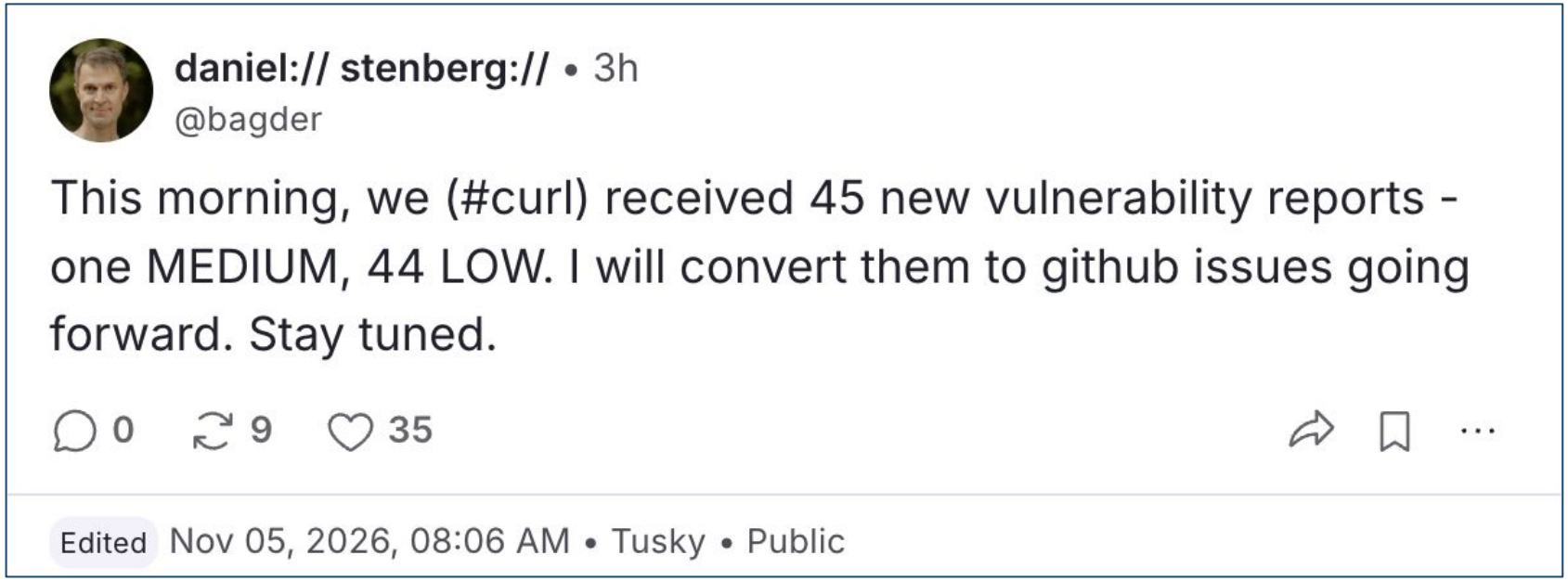
PACKAGE	UBUNTU RELEASE	STATUS
linux	26.10 stonking	Vulnerable work in progress in 7.3.0-10.10
	26.04 LTS resolute	Vulnerable work in progress in 7.0.0-43.43
	24.04 LTS noble	Vulnerable work in progress in 6.8.0-149.149
	22.04 LTS jammy	Vulnerable work in progress in 5.15.0-203.213
	20.04 LTS focal	Vulnerable
	18.04 LTS bionic	Vulnerable
	16.04 LTS xenial	Vulnerable

Different projects have been affected differently

curl has been flooded.

Others seem to be quietly queueing up fixes.

I'm assuming some projects are unaffected? Not everyone has millions of lines of attack surface.



What if we get an absolute massive flood of reports?

This is well underway

Aisle research sent ~200 reports our way earlier this year.

There's a long tail of reports from others.

We're looking for issues ourselves, and one dev has found quite a few.

This might require require policy adjustments

From the curl vulnerability disclosure policy:

Low/Medium severity issues are public

For all reported vulnerabilities with severity LOW or MEDIUM, the reporter is asked to instead submit it publicly in the curl issue tracker on GitHub, or even better as a pull-request. These issues are then not managed any further privately on HackerOne.

If in doubt of the severity, starting out on HackerOne is fine.

Asking the reporter to instead submit the issue on GitHub does not necessarily imply that we accept it as a vulnerability, but we might. It might also be a plain bug. It should be treated as a "let's continue the conversation over there in the public" request.

If a reported issue is deemed to be severity HIGH or CRITICAL, it should remain on HackerOne to be handled privately until a pending releases can be shipped that includes a fix for the problem.

We might have to make adjustments as well.

Expectation mismatches

Assuming that any given OSS project can handle a sudden workload surge isn't realistic.

Open source project with paid teams *might* be able to handle a 100x load.

Volunteer projects with fewer resources definitely can't.



What might help

What works for us

Aggressively look for bugs, otherwise you're in for a world of hurt.

What works for us

Aggressively look for bugs, otherwise you're in for a world of hurt.

Secure your software, not your ego.

What works for us

Aggressively look for bugs, otherwise you're in for a world of hurt.

Secure your software, not your ego.

Personal interaction still matters

Looking forward

If you're a Wireshark user: We're going to keep working to keep Wireshark secure. AI is one of many tools in that particular toolbox.

If you're a security researcher: We're happy to accept vulnerability reports, even during storms. Please read SECURITY.md.

We're slowly moving from reacting to this flood to being more preemptive and proactive.

Thank You

