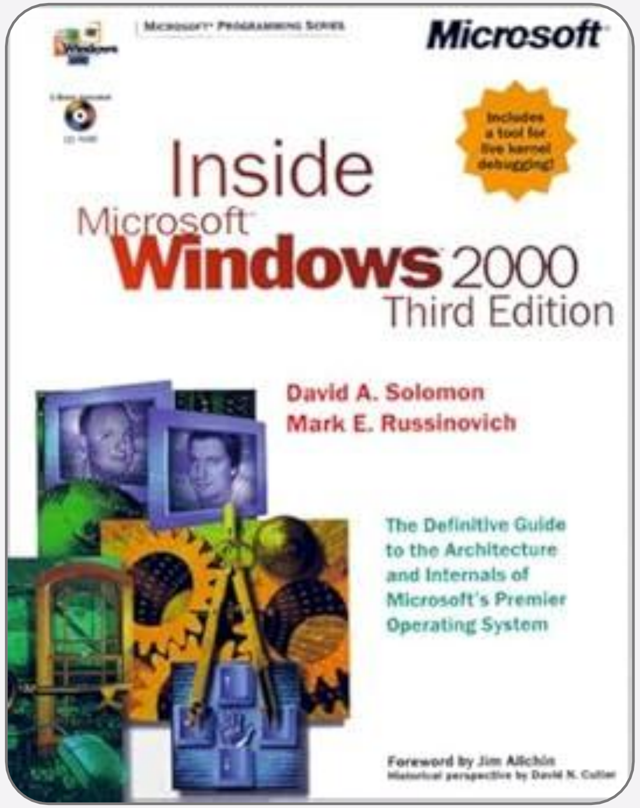
Microsoft

Toward a Secure and Sustainable Open Source Supply Chain

Mark Russinovich
CTO, Deputy CISO and Technical Fellow,
Microsoft Azure

🦋 @markrussinovich
@ @mrussinovich



Microsoft Programming Series

Microsoft

Includes a tool for live kernel debugging!

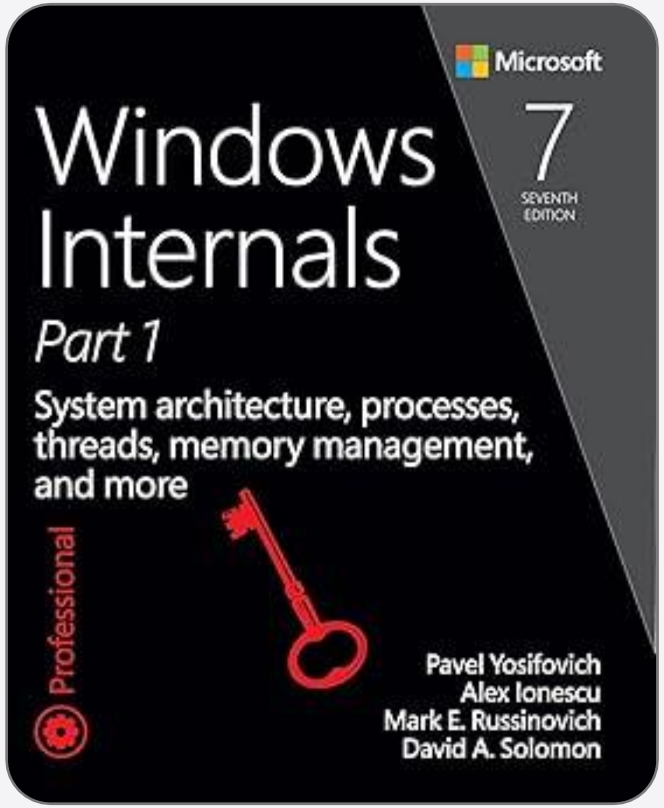
Inside Microsoft Windows 2000

Third Edition

David A. Solomon
Mark E. Russinovich

The Definitive Guide to the Architecture and Internals of Microsoft's Premier Operating System

Foreword by Jim Allchin
Historical perspective by David N. Cutler



Microsoft

Windows Internals

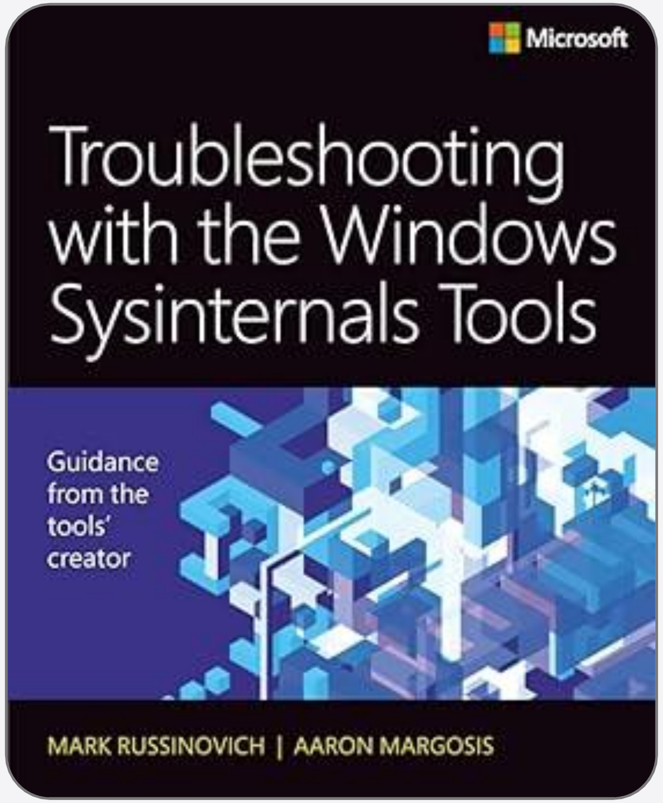
Part 1

System architecture, processes, threads, memory management, and more

7
SEVENTH EDITION

Professional

Pavel Yosifovich
Alex Ionescu
Mark E. Russinovich
David A. Solomon

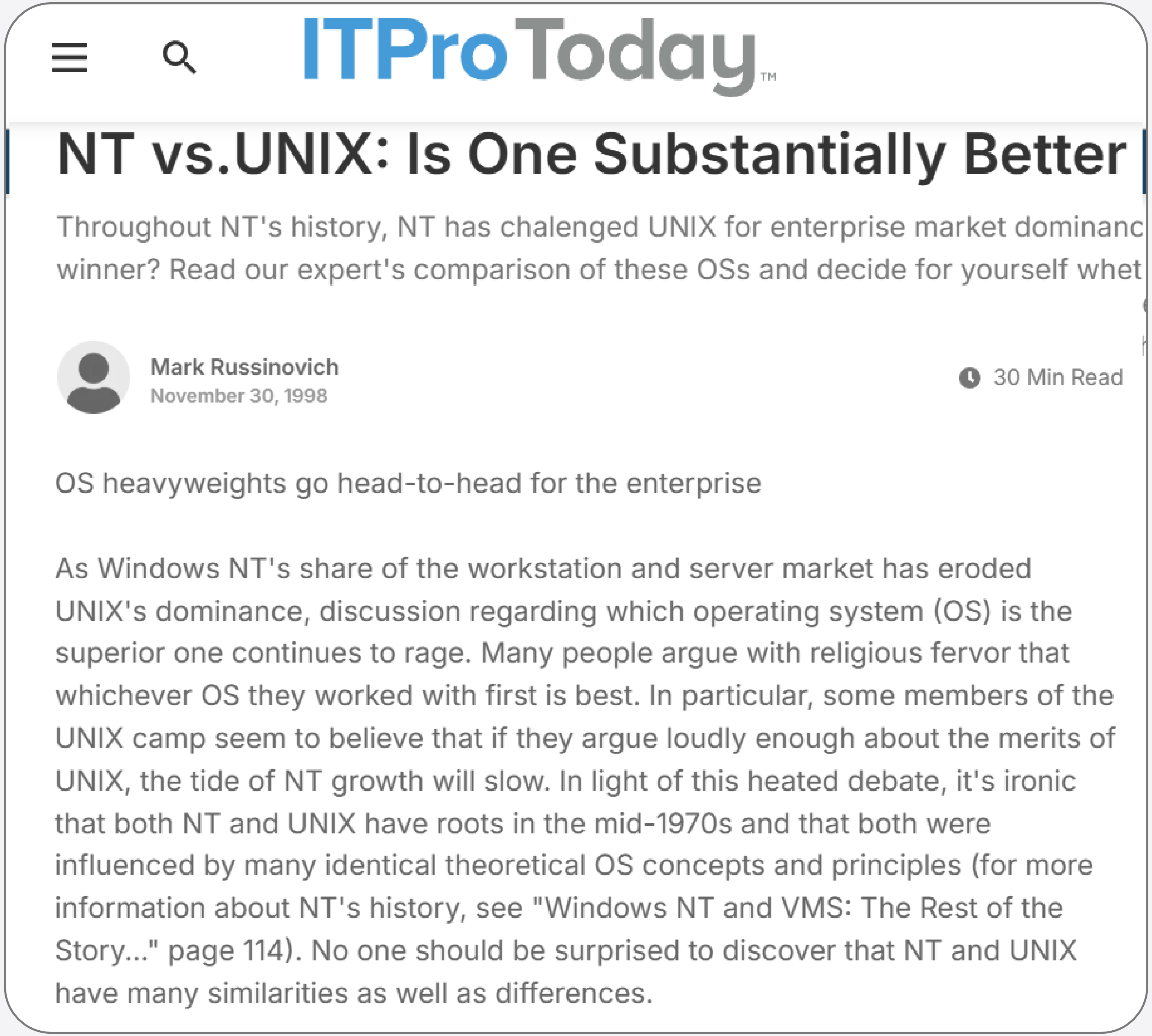


Microsoft

Troubleshooting with the Windows Sysinternals Tools

Guidance from the tools' creator

MARK RUSSINOVICH | AARON MARGOSIS



ITProToday

NT vs.UNIX: Is One Substantially Better

Throughout NT's history, NT has chalenged UNIX for enterprise market dominanc
winner? Read our expert's comparison of these OSs and decide for yourself whet

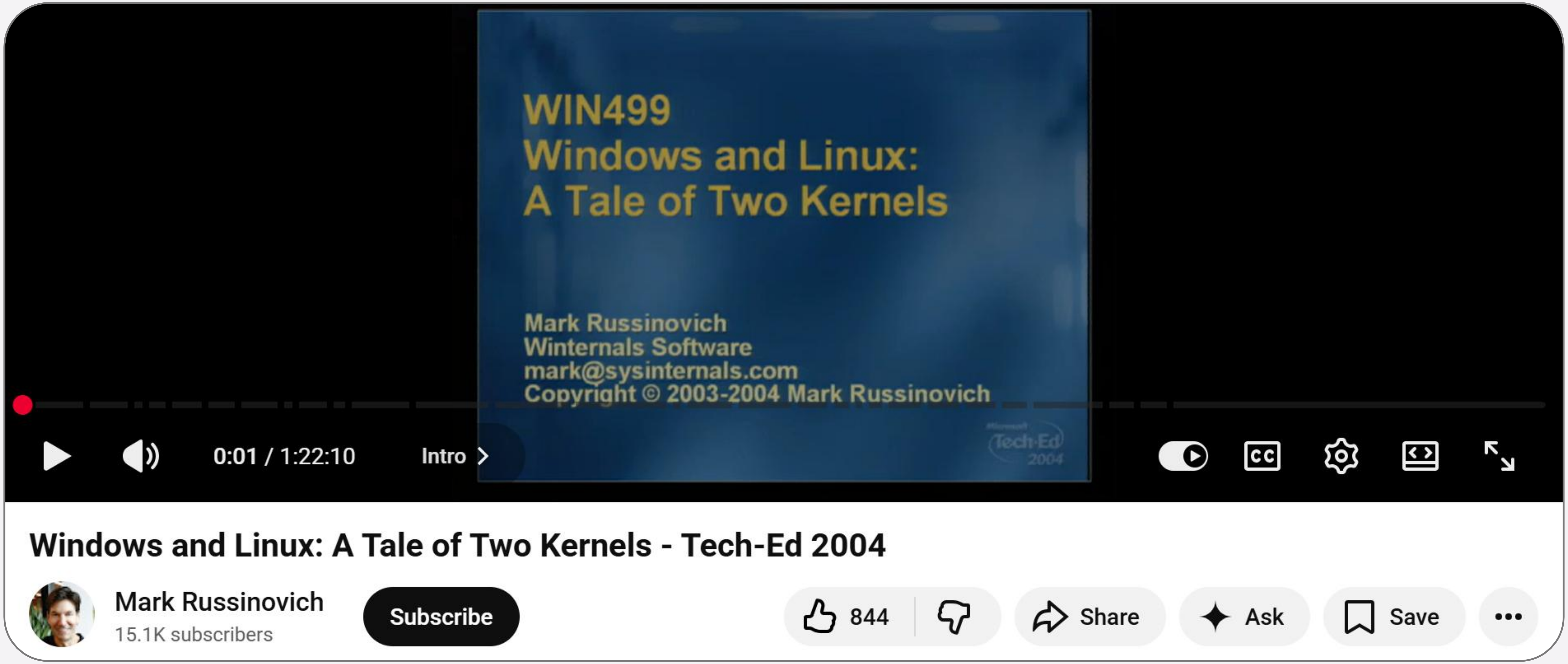
Mark Russinovich

November 30, 1998

30 Min Read

OS heavyweights go head-to-head for the enterprise

As Windows NT's share of the workstation and server market has eroded UNIX's dominance, discussion regarding which operating system (OS) is the superior one continues to rage. Many people argue with religious fervor that whichever OS they worked with first is best. In particular, some members of the UNIX camp seem to believe that if they argue loudly enough about the merits of UNIX, the tide of NT growth will slow. In light of this heated debate, it's ironic that both NT and UNIX have roots in the mid-1970s and that both were influenced by many identical theoretical OS concepts and principles (for more information about NT's history, see "Windows NT and VMS: The Rest of the Story..." page 114). No one should be surprised to discover that NT and UNIX have many similarities as well as differences.

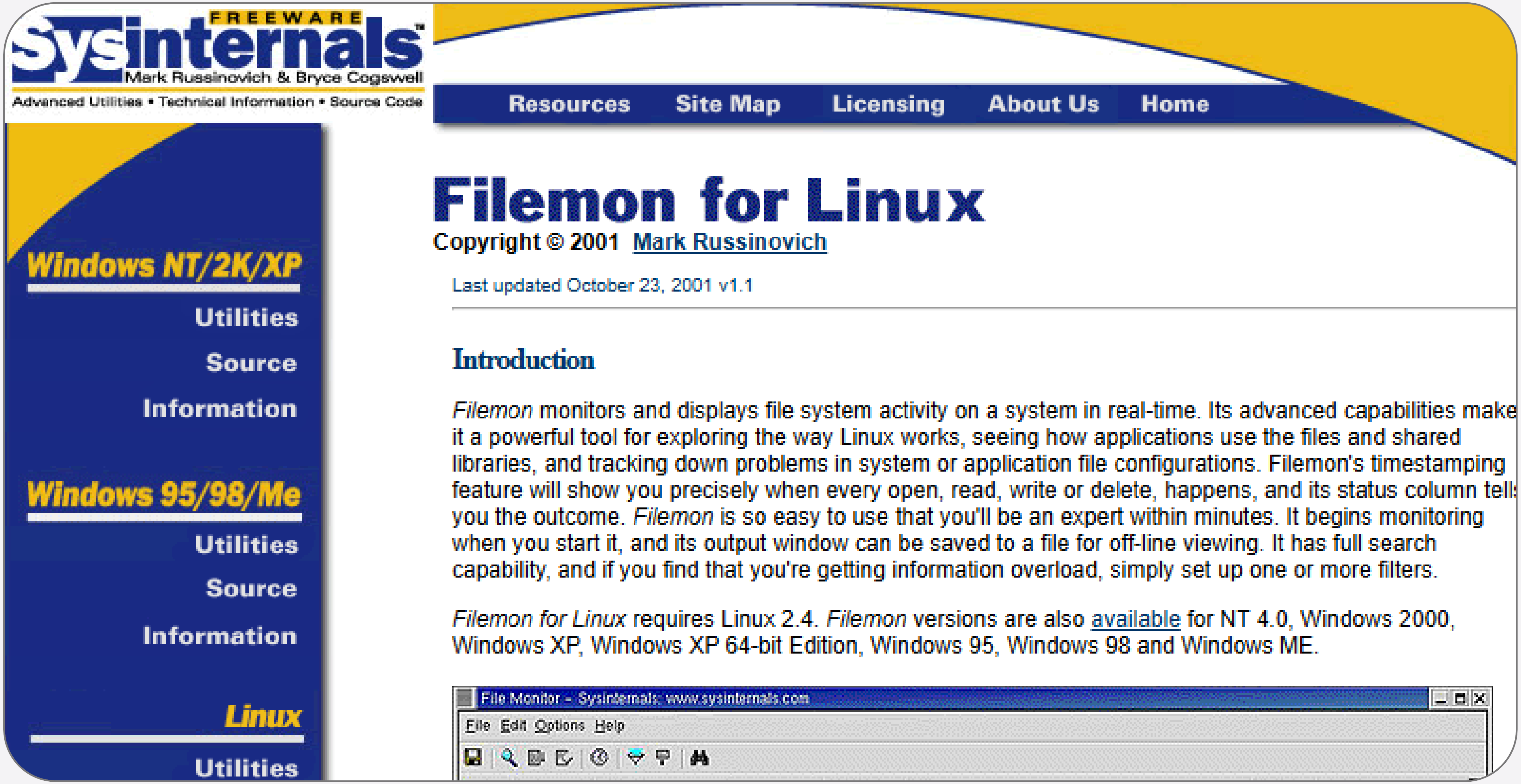


WIN499
Windows and Linux:
A Tale of Two Kernels

Mark Russinovich
Winternals Software
mark@sysinternals.com
Copyright © 2003-2004 Mark Russinovich

Windows and Linux: A Tale of Two Kernels - Tech-Ed 2004

Mark Russinovich
15.1K subscribers



FREEWARE Sysinternals™
Mark Russinovich & Bryce Cogswell
Advanced Utilities • Technical Information • Source Code

Resources | Site Map | Licensing | About Us | Home

Windows NT/2K/XP
- Utilities
- Source
- Information

Windows 95/98/Me
- Utilities
- Source
- Information

Linux
- Utilities

Filemon for Linux

Copyright © 2001 Mark Russinovich

Last updated October 23, 2001 v1.1

Introduction

Filemon monitors and displays file system activity on a system in real-time. Its advanced capabilities make it a powerful tool for exploring the way Linux works, seeing how applications use the files and shared libraries, and tracking down problems in system or application file configurations. Filemon's timestamping feature will show you precisely when every open, read, write or delete, happens, and its status column tells you the outcome. *Filemon* is so easy to use that you'll be an expert within minutes. It begins monitoring when you start it, and its output window can be saved to a file for off-line viewing. It has full search capability, and if you find that you're getting information overload, simply set up one or more filters.

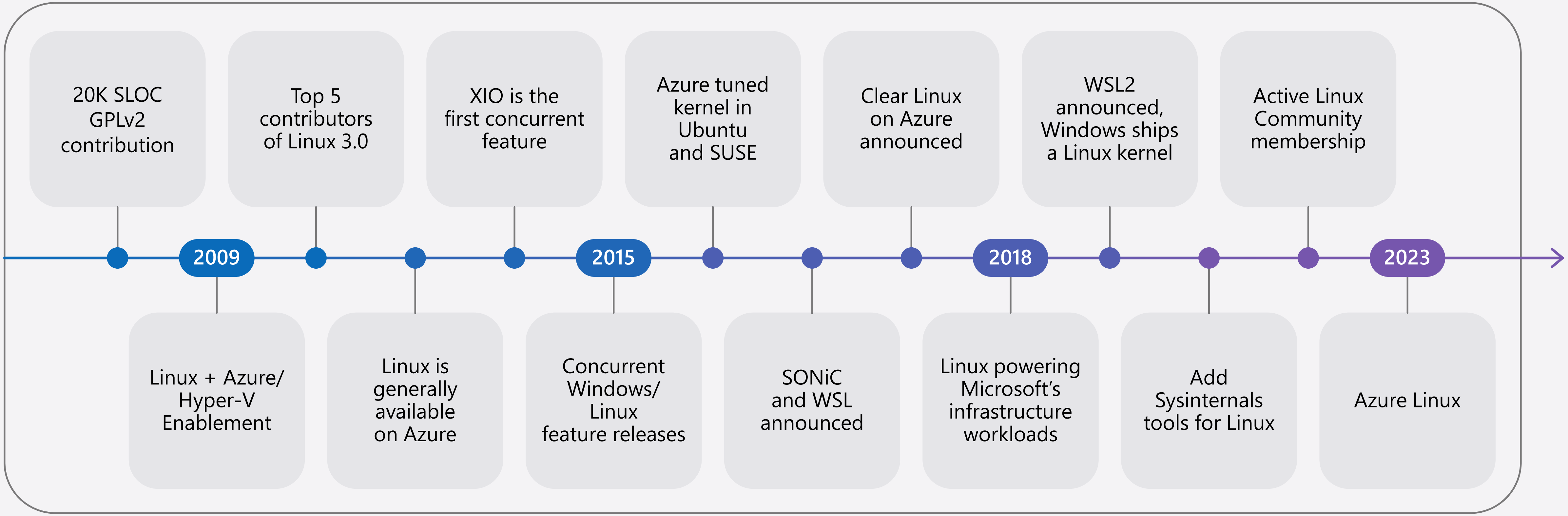
Filemon for Linux requires Linux 2.4. *Filemon* versions are also available for NT 4.0, Windows 2000, Windows XP, Windows XP 64-bit Edition, Windows 95, Windows 98 and Windows ME.

Mark
Microsoft
Microsoft ♥ Linux

Me
Linus Torvalds
Jim Zemlin

Microsoft's Linux Open Source Journey

Microsoft's Linux Open Source Journey



20K SLOC GPLv2 contribution

2009

Linux + Azure/ Hyper-V Enablement

Top 5 contributors of Linux 3.0

Linux is generally available on Azure

XIO is the first concurrent feature

2015

Concurrent Windows/ Linux feature releases

Azure tuned kernel in Ubuntu and SUSE

SONiC and WSL announced

Clear Linux on Azure announced

2018

Linux powering Microsoft's infrastructure workloads

WSL2 announced, Windows ships a Linux kernel

Add Sysinternals tools for Linux

Active Linux Community membership

2023

Azure Linux

Linux in Azure

>65%

of customer compute cores are Linux

>60%

of Azure marketplace images are Linux-based

ChatGPT on Azure

900 million
weekly active users

The fastest growing
app in history

Azure
GPU VMs

Azure
Kubernetes Service

Azure
Cosmos DB

Azure
PostgreSQL

Azure
Storage

Open source is at the core of Microsoft

Microsoft's Open Source Software Policy

Company-wide documented policy to make it easy for Microsoft developers to use open source

Using open source

Contributing to open source

Releasing open source software

We share some of our policy publicly on the opensource.microsoft.com site under "Our programs"

200k

open source components
every month

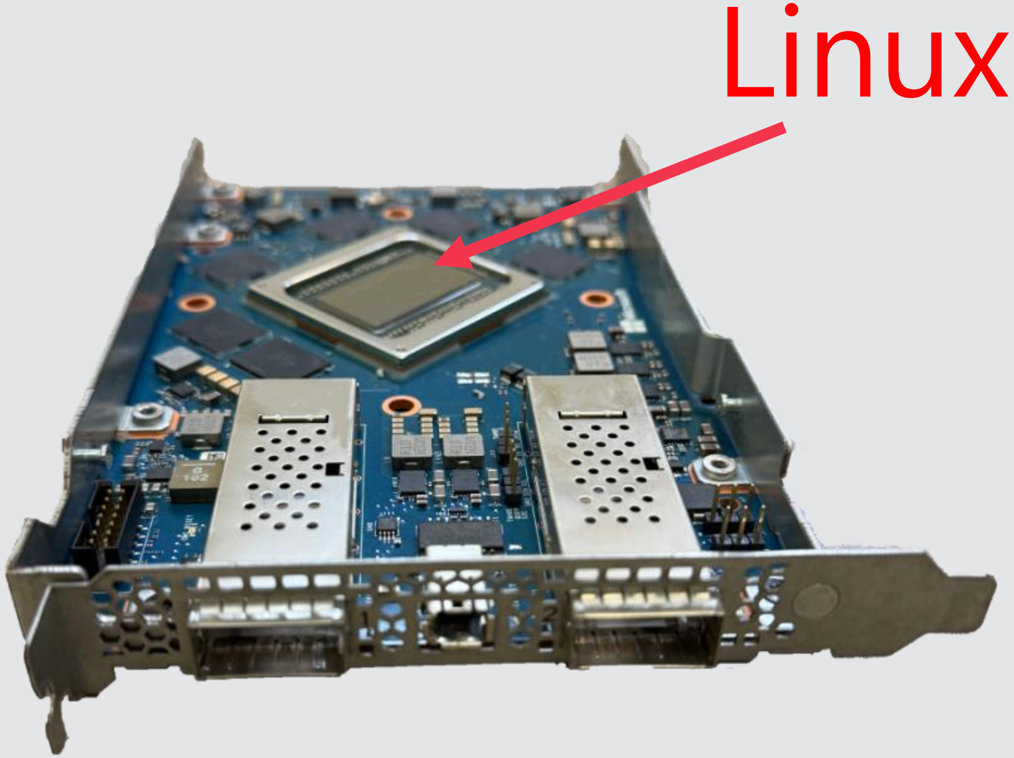
11.8M

places in use

Microsoft 365’s COSMIC Runs on Azure Kubernetes Service

One of the largest Kubernetes clusters in the world
Runs Microsoft 365’s cloud services on millions of cores

Azure Boost



Linux

Microsoft's Open Source Software Policy

Company-wide documented policy to make it easy for Microsoft developers to use open source

Using open source

Contributing to open source

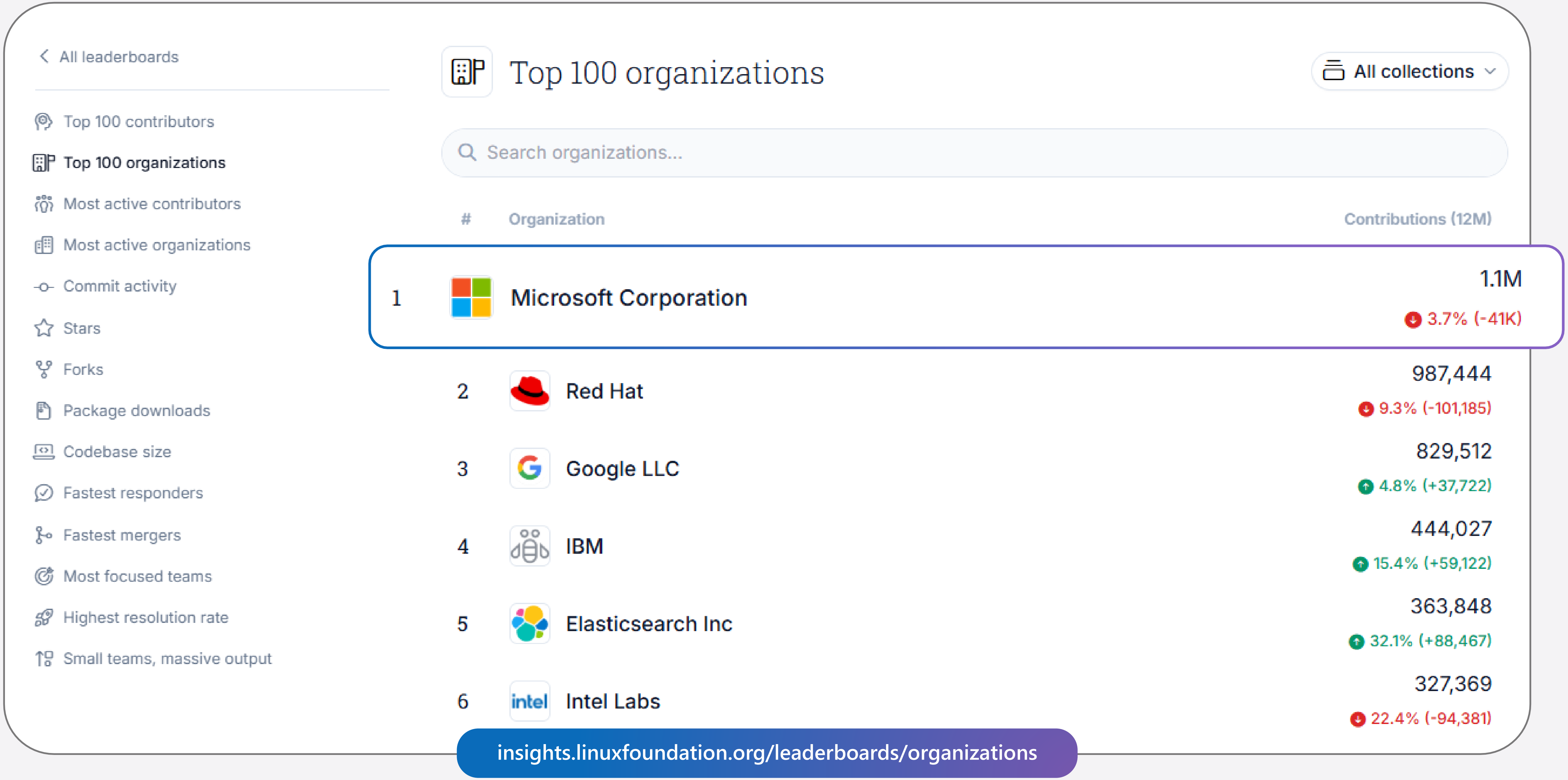
Releasing open source software

We share some of our policy publicly on the opensource.microsoft.com site under "Our programs"

3M

contributions to open source in the past 3 months

Counted by PR reviews/creation/comments, team discussions, issue creation/comments, commit comments to an open source or public repo.



Top 100 organizations

#	Organization	Contributions (12M)
1	Microsoft Corporation	1.1M 3.7% (-41K)
2	Red Hat	987,444 9.3% (-101,185)
3	Google LLC	829,512 4.8% (+37,722)
4	IBM	444,027 15.4% (+59,122)
5	Elasticsearch Inc	363,848 32.1% (+88,467)
6	Intel Labs	327,369 22.4% (-94,381)

insights.linuxfoundation.org/leaderboards/organizations

Microsoft's Open Source Software Policy

Company-wide documented policy to make it easy for Microsoft developers to use open source

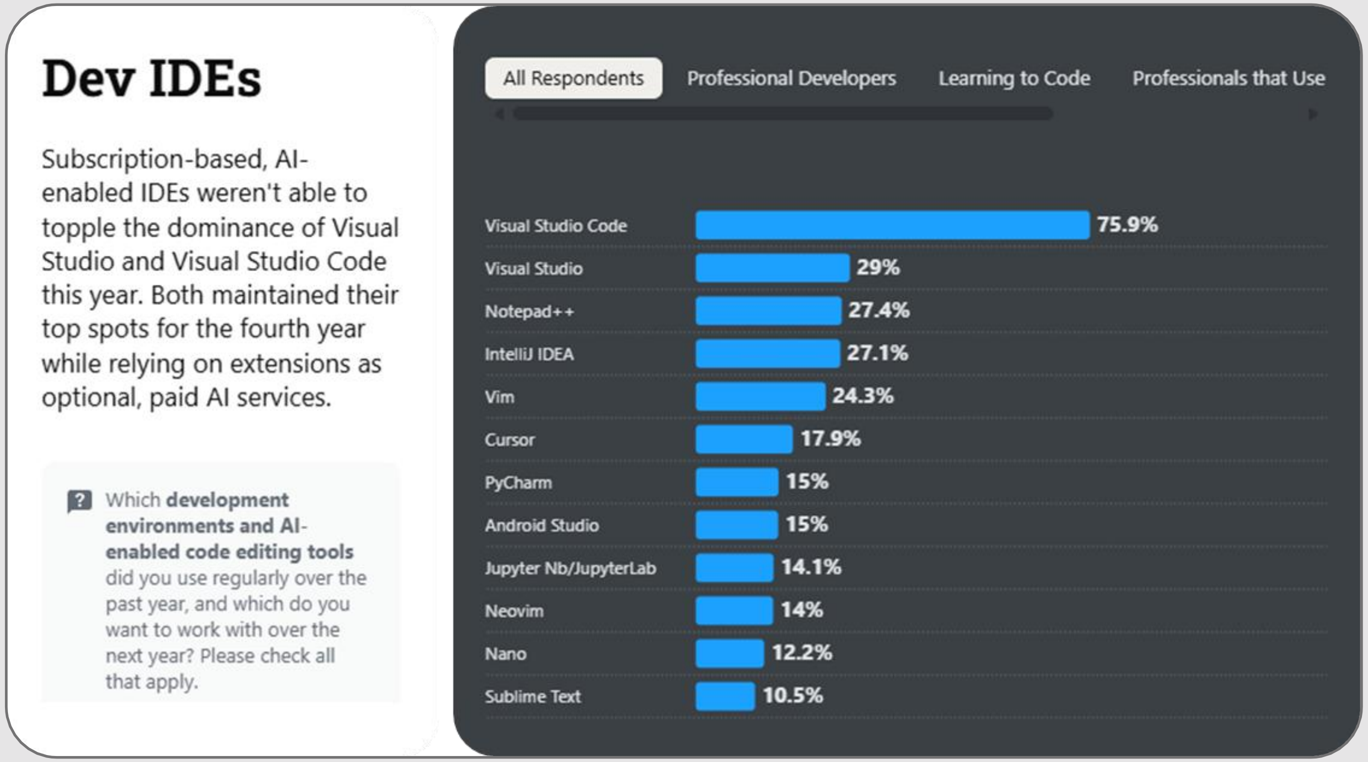
Using
open source

Contributing
to open source

**Releasing open
source software**

We share some of our policy publicly on the opensource.microsoft.com site under "Our programs"

Visual Studio Code



Dev IDEs

Subscription-based, AI-enabled IDEs weren't able to topple the dominance of Visual Studio and Visual Studio Code this year. Both maintained their top spots for the fourth year while relying on extensions as optional, paid AI services.

Which **development environments and AI-enabled code editing tools** did you use regularly over the past year, and which do you want to work with over the next year? Please check all that apply.

survey.stackoverflow.co/2025/

More for developers

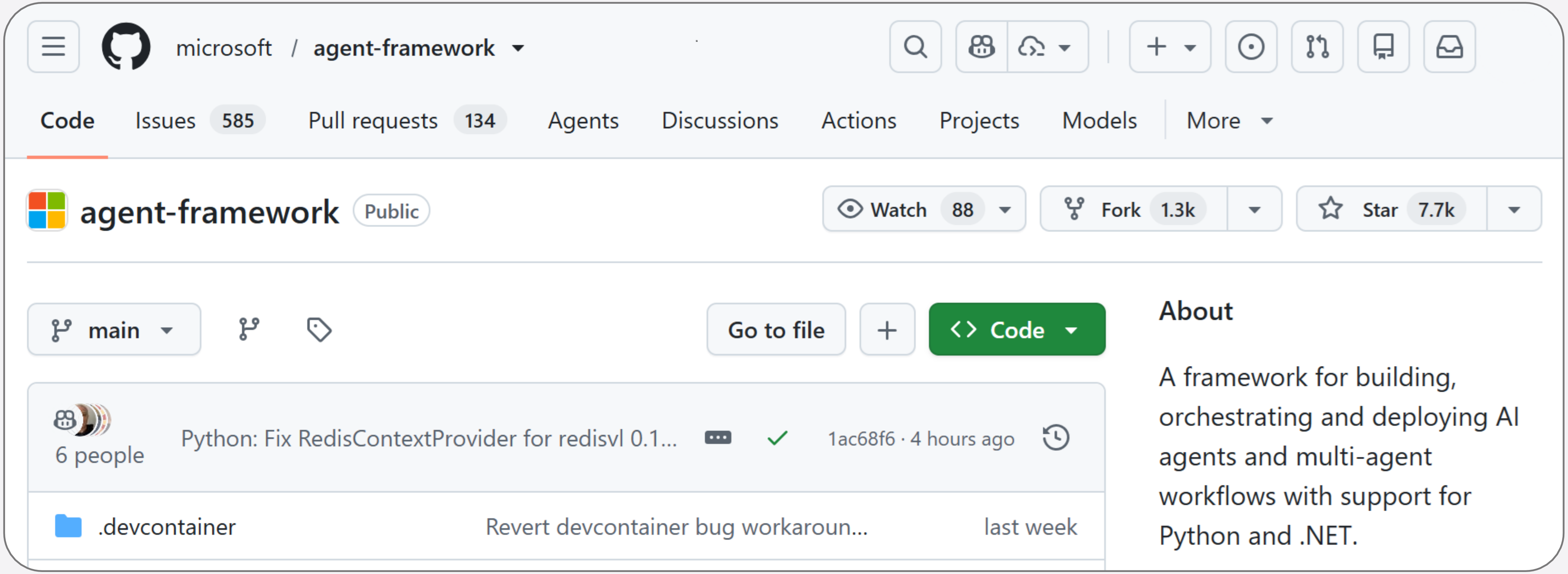
.NET

TypeScript

PowerShell

Microsoft Build of OpenJDK™

Microsoft Agent Framework



agent-framework Public

6 people Python: Fix RedisContextProvider for redisvl 0.1... 1ac68f6 · 4 hours ago

.devcontainer Revert devcontainer bug workaroun... last week

About

A framework for building, orchestrating and deploying AI agents and multi-agent workflows with support for Python and .NET.

Azure Incubations

Mission

Partner across Microsoft and the open-source community to explore and deliver industry-changing products

KEDA

GRADUATED
Aug 2023

dapr

GRADUATED
Nov 2024

copa

SANDBOX
Sept 2023

radius

SANDBOX
Apr 2024

drasi

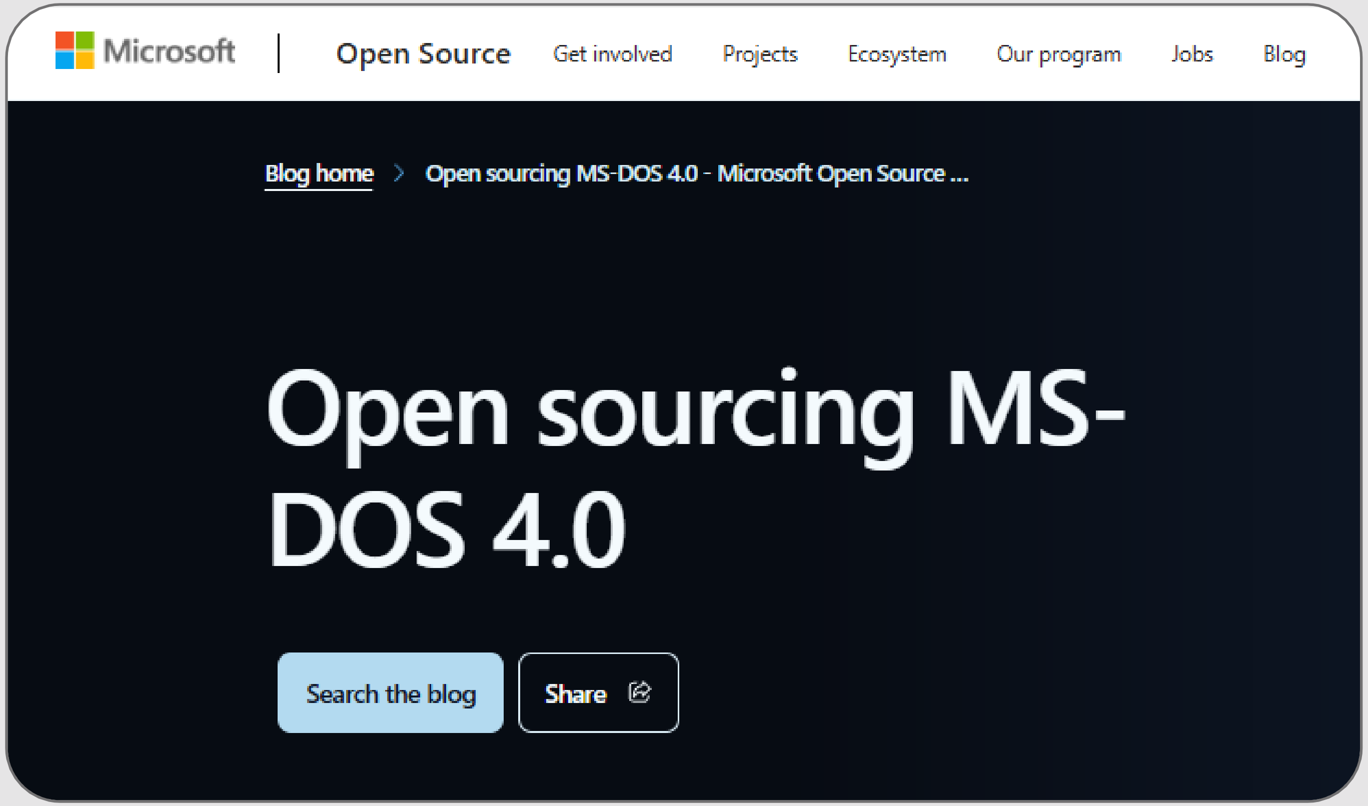
SANDBOX
Jan 2025

CLOUD NATIVE
COMPUTING FOUNDATION

Sysinternals for Linux

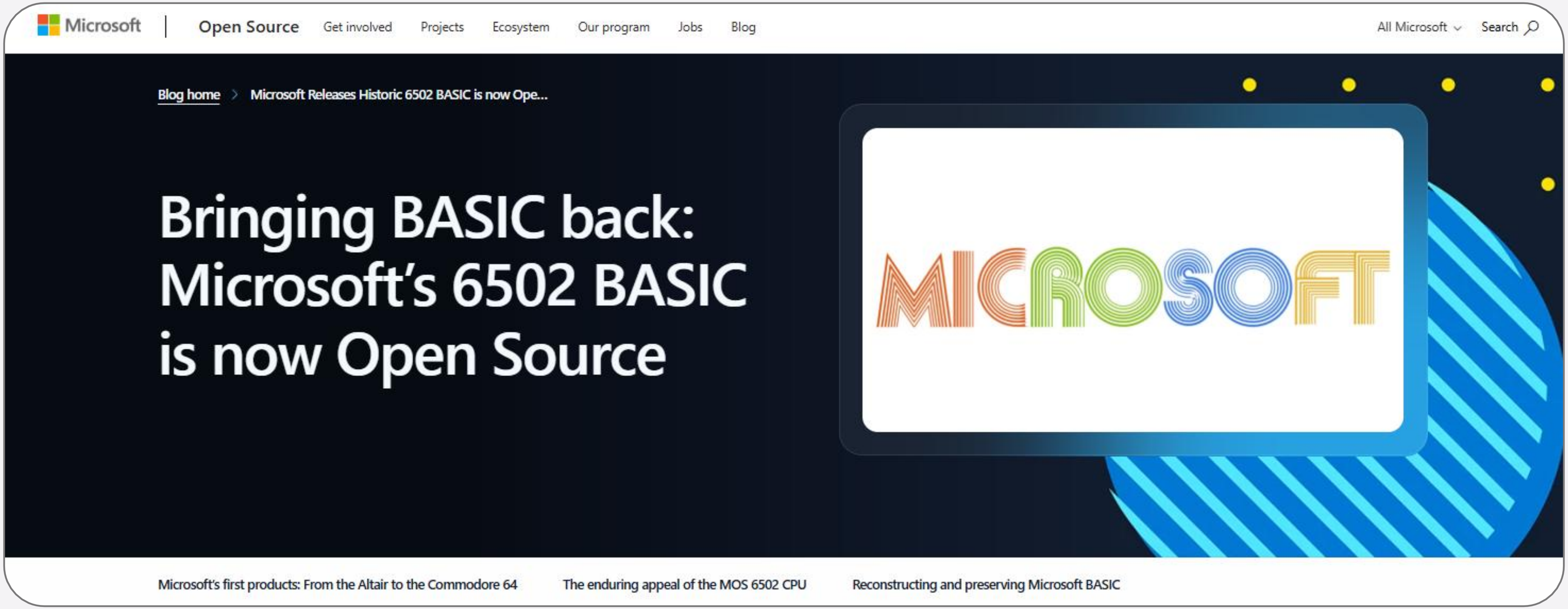
- ProcDump
- Sysmon
- Procmon
- jcd
- SysinternalsEBPF

Written almost 45 years ago entirely in 8086 assembly!



Open sourcing MS-DOS 4.0

Open sourcing MS-DOS 4.0 - Microsoft Open Source Blog – April 25, 2024



Blog home > Microsoft Releases Historic 6502 BASIC is now Ope...

Bringing BASIC back: Microsoft's 6502 BASIC is now Open Source

Microsoft's first products: From the Altair to the Commodore 64

The enduring appeal of the MOS 6502 CPU

Reconstructing and preserving Microsoft BASIC

The whole world depends on Open Source Software

More concretely…

97%	of the codebases contained open source
70%	of scanned code had its origin in open source
900	Average number of OSS components found per application
64%	of OSS components were transitive dependencies

Source: blackduck.com/content/dam/black-duck/en-us/reports/rep-ossra.pdf
sonatype.com/hubfs/1-2025_Website-Assets/SSCR_2025/SSCR_2026_final.pdf

Package Registry growth

Open Source Package registries like PyPI, crates.io, RubyGems are critical infrastructure for software development

10 Trillion
downloads per year

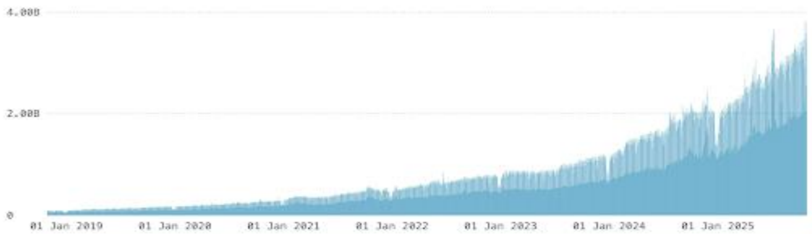
Yet this entire ecosystem runs largely on donations and in-kind infrastructure

Increased attacks over the years (typosquatting, malware packages)

PyPi

~3 Billion Downloads per day
~100+ Billion Downloads per Month
Hosts **~860K packages**

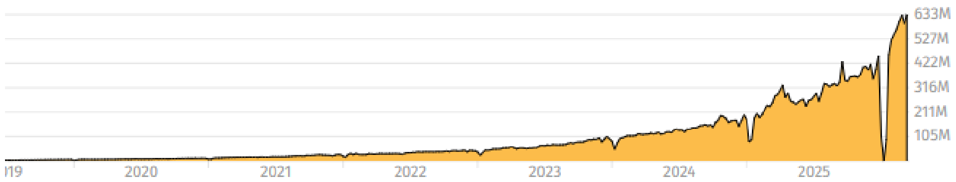
source: pypistats.org



Crates.io

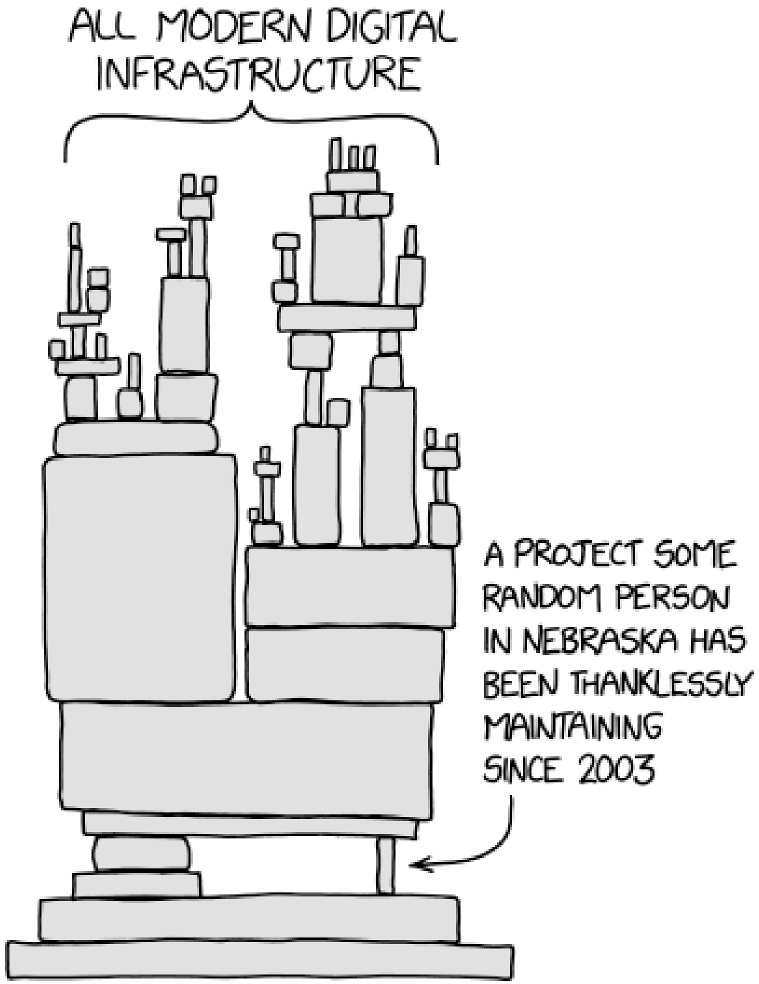
Download growth 2.2x per year
~730 M download per day

source: lib.rs/stats



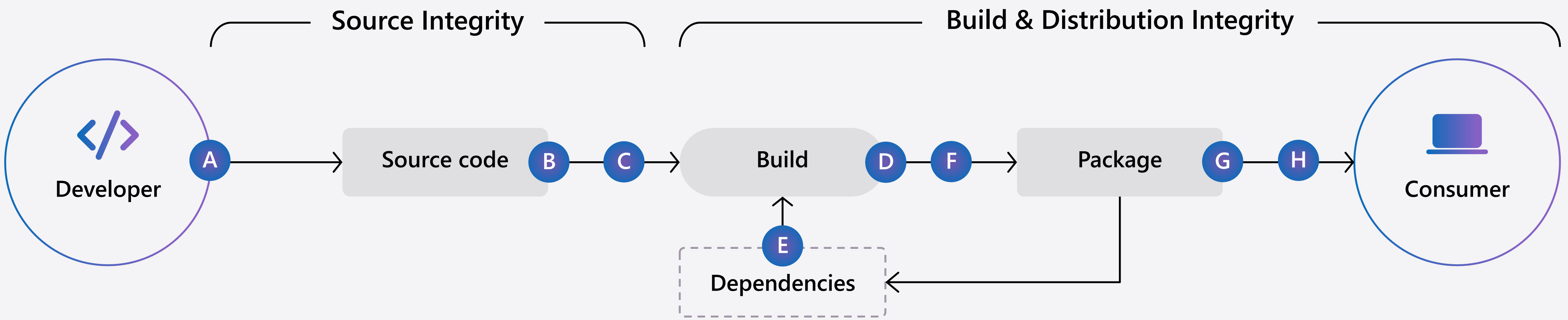
Daily downloads since Rust 1.0, 7-day average

Attackers treat open source as a delivery channel


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

Source: https://xkcd.com/2347

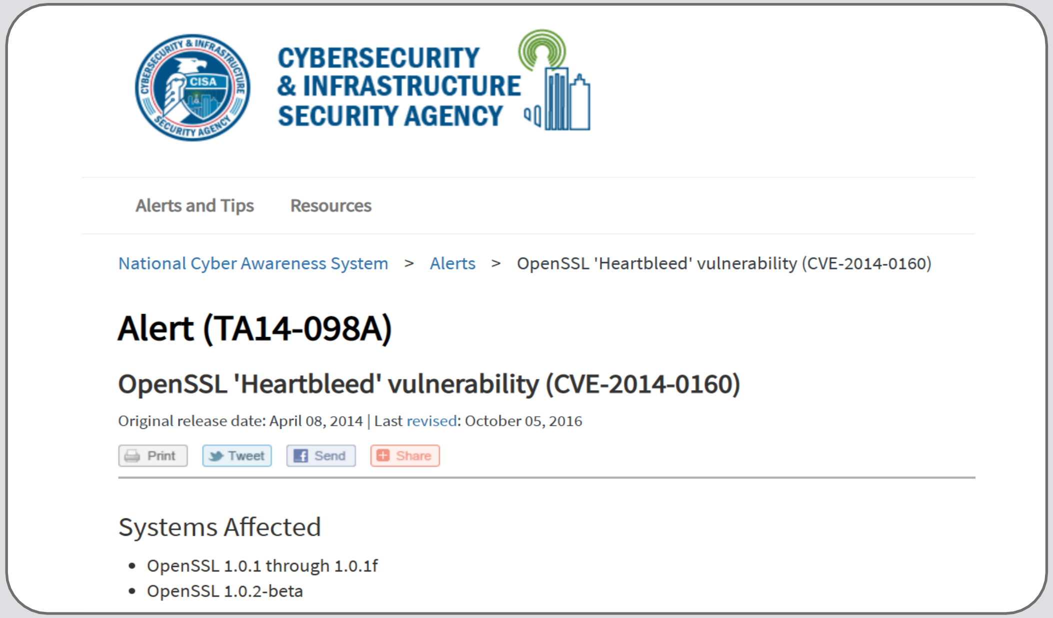
The open source supply chain



- A Bypassed code review
- B Compromised source control system
- C Modified code after source control
- D Compromised build platform
- E Using a bad dependency
- F Bypassed CI/CD
- G Compromised package repo
- H Using a bad package

OpenSSL

2014: Heartbleed


CYBERSECURITY
& INFRASTRUCTURE
SECURITY AGENCY
Alerts and Tips
Resources
National Cyber Awareness System > Alerts > OpenSSL 'Heartbleed' vulnerability (CVE-2014-0160)
Print
Tweet
Send
Share

Alert (TA14-098A)

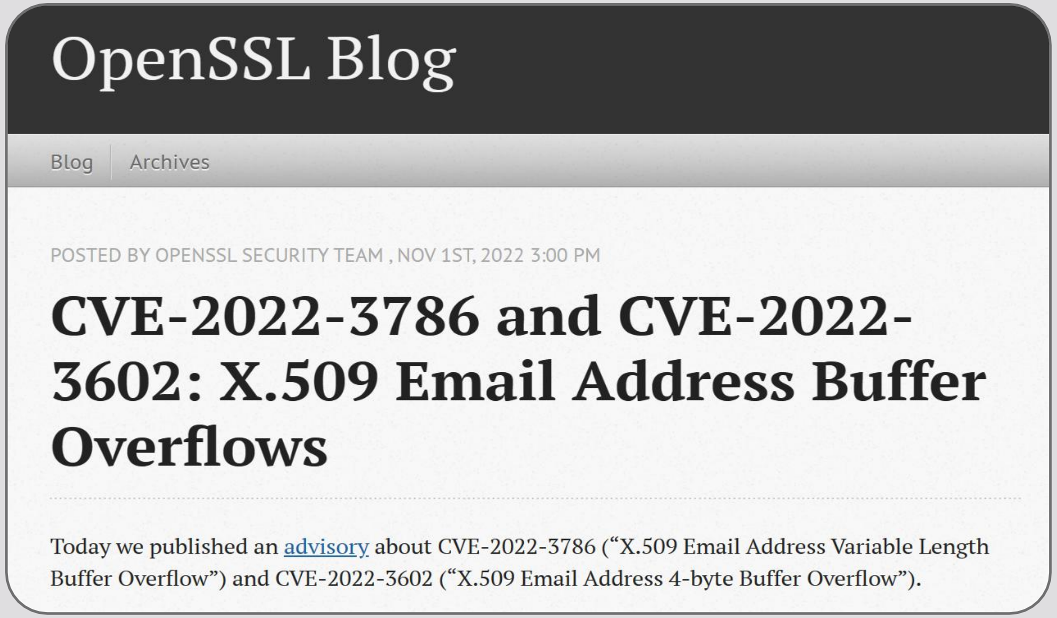
OpenSSL 'Heartbleed' vulnerability (CVE-2014-0160)

Original release date: April 08, 2014 | Last revised: October 05, 2016

Systems Affected

- OpenSSL 1.0.1 through 1.0.1f
- OpenSSL 1.0.2-beta

2022: Buffer overflows

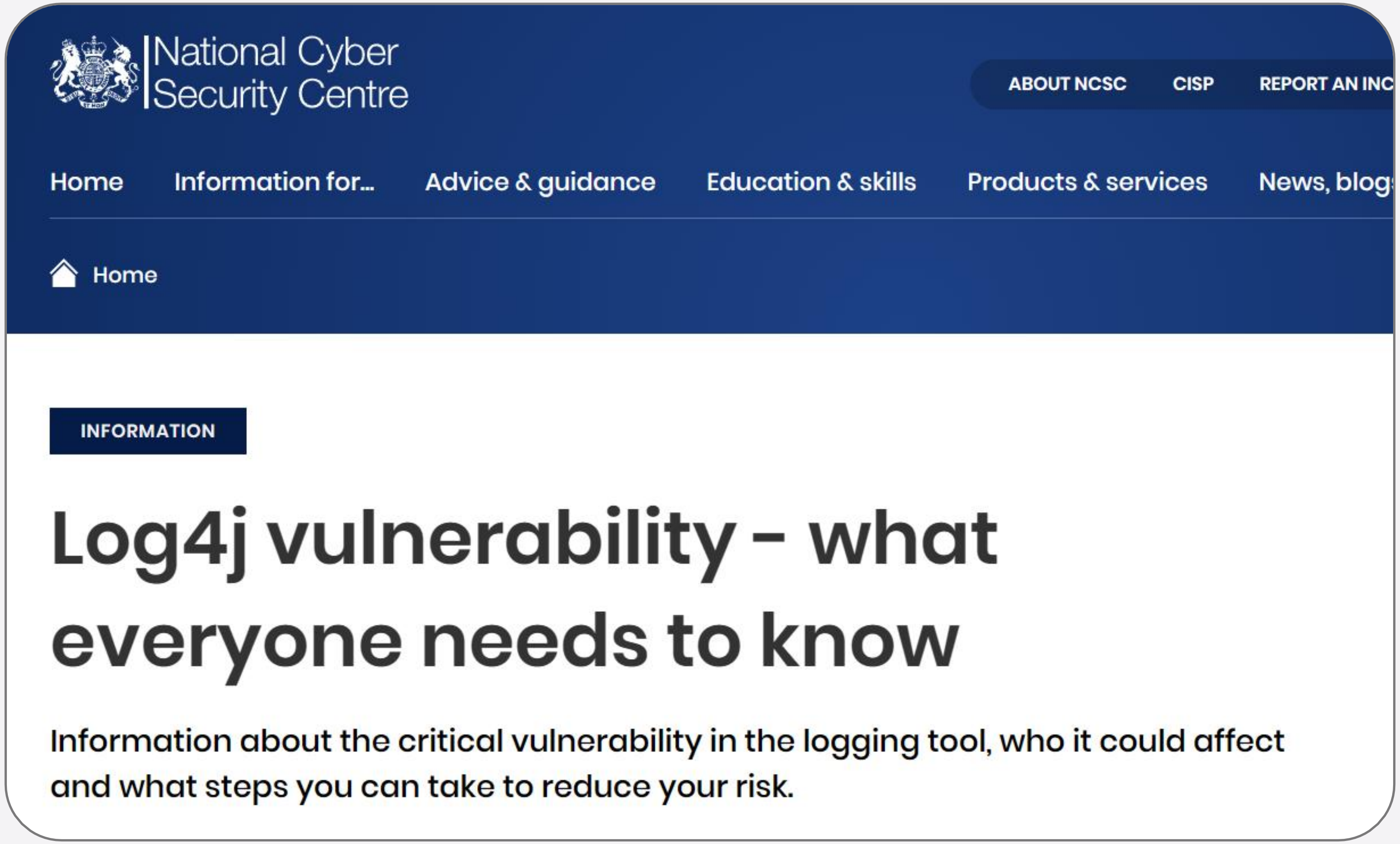


POSTED BY OPENSSL SECURITY TEAM , NOV 1ST, 2022 3:00 PM

CVE-2022-3786 and CVE-2022-3602: X.509 Email Address Buffer Overflows

Today we published an advisory about CVE-2022-3786 (“X.509 Email Address Variable Length Buffer Overflow”) and CVE-2022-3602 (“X.509 Email Address 4-byte Buffer Overflow”).

Log4j

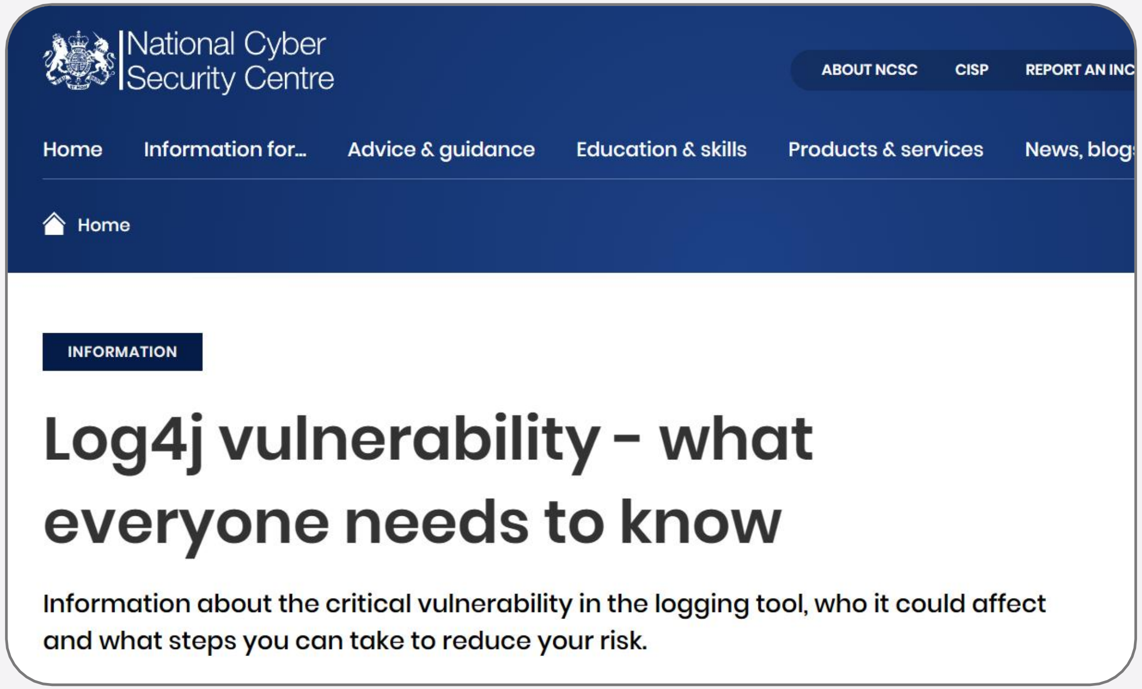


Log4j vulnerability - what everyone needs to know

Information about the critical vulnerability in the logging tool, who it could affect and what steps you can take to reduce your risk.

December, 2021

2025



National Cyber Security Centre

INFORMATION

Log4j vulnerability - what everyone needs to know

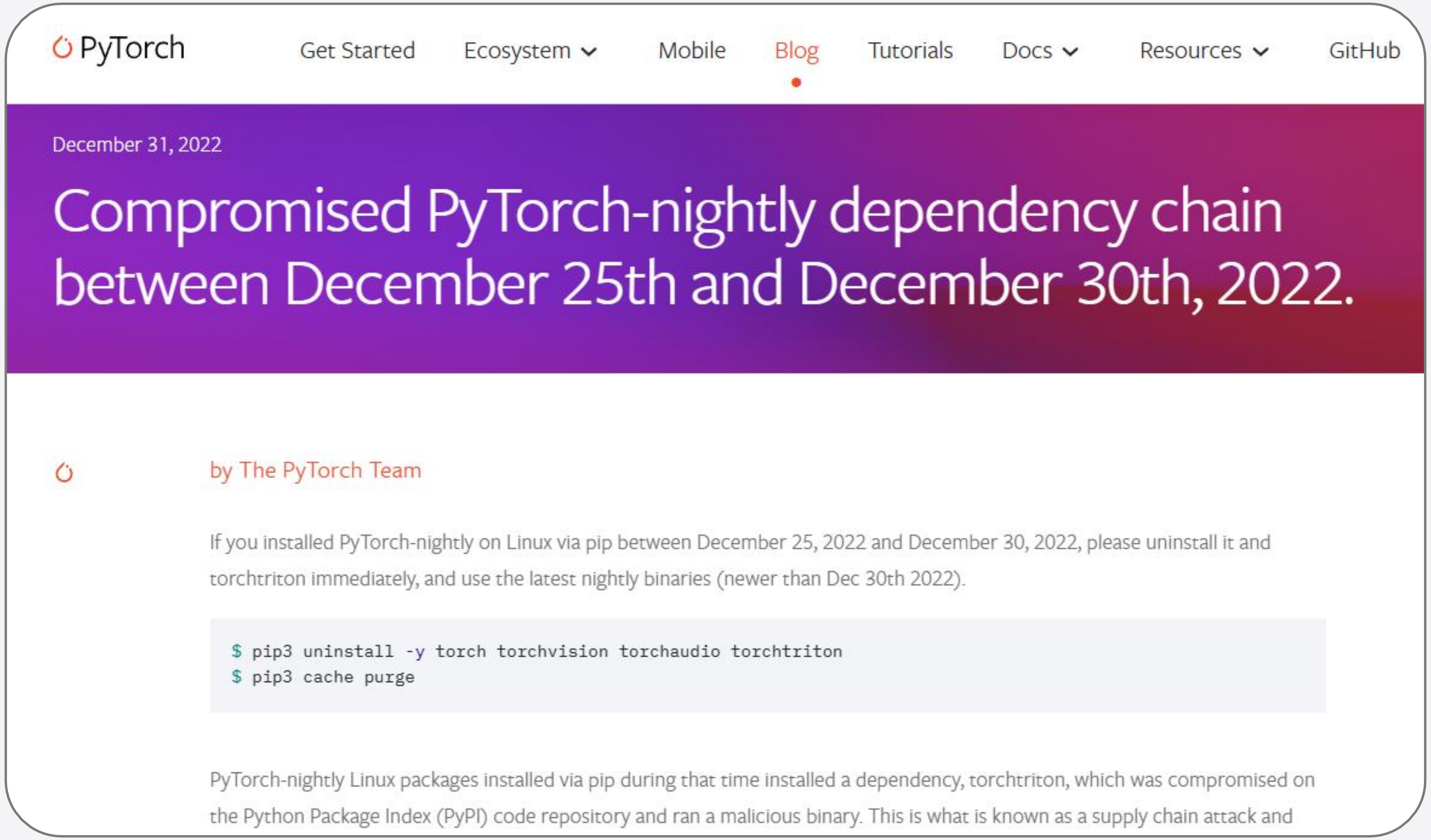
Information about the critical vulnerability in the logging tool, who it could affect and what steps you can take to reduce your risk.

4 years later

In 2025 alone, developers downloaded more than **42 million vulnerable versions of log4j**, representing **13% of all log4j downloads worldwide.**

sonatype.com/whitepapers/the-persistence-of-open-source-vulnerabilities

Pytorch



December 31, 2022

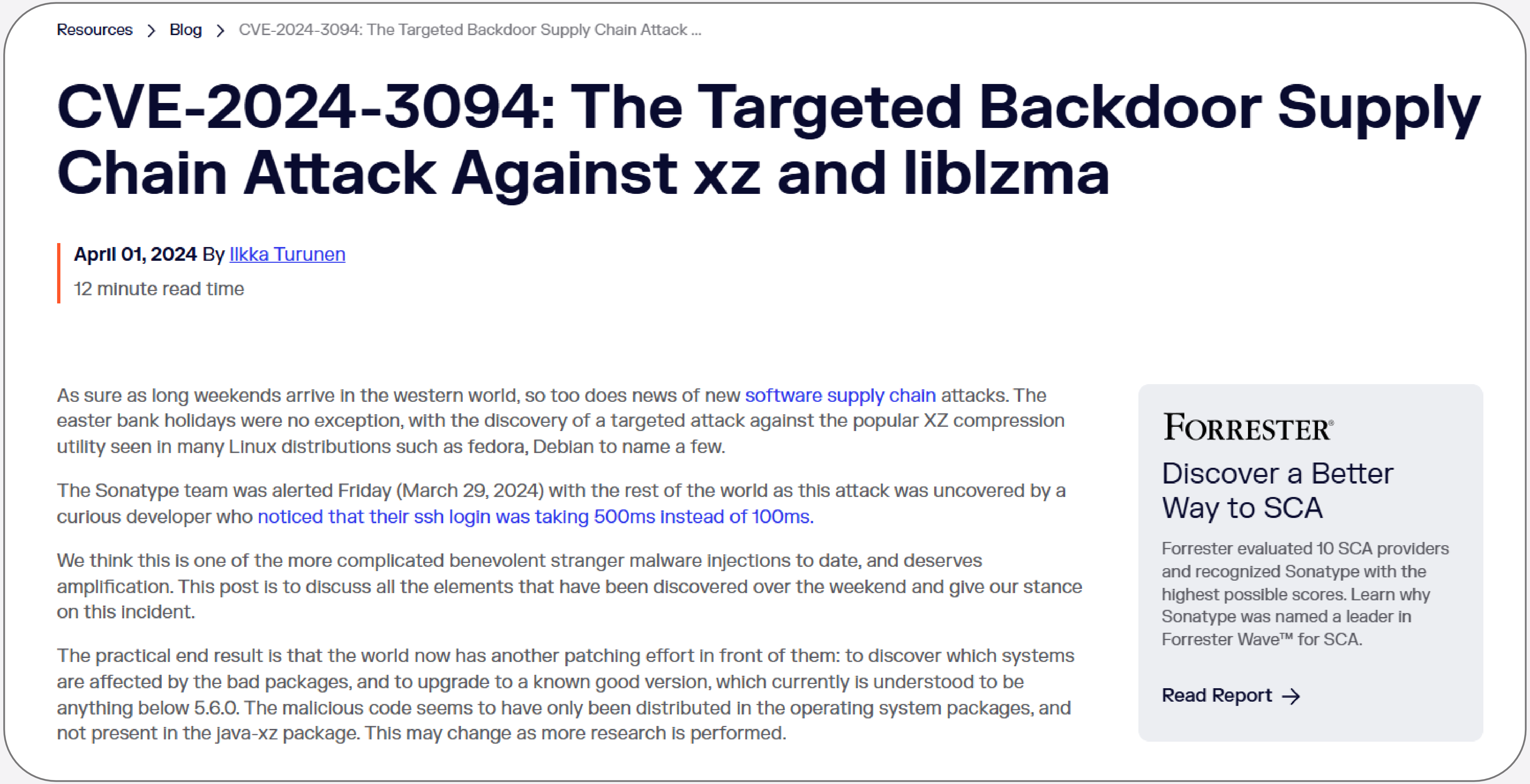
Compromised PyTorch-nightly dependency chain between December 25th and December 30th, 2022.

by The PyTorch Team

If you installed PyTorch-nightly on Linux via pip between December 25, 2022 and December 30, 2022, please uninstall it and torchtriton immediately, and use the latest nightly binaries (newer than Dec 30th 2022).

```
$ pip3 uninstall -y torch torchvision torchaudio torchtriton
$ pip3 cache purge
```

PyTorch-nightly Linux packages installed via pip during that time installed a dependency, torchtriton, which was compromised on the Python Package Index (PyPI) code repository and ran a malicious binary. This is what is known as a supply chain attack and



Resources > Blog > CVE-2024-3094: The Targeted Backdoor Supply Chain Attack ...

CVE-2024-3094: The Targeted Backdoor Supply Chain Attack Against xz and liblzma

April 01, 2024 By Ilkka Turunen

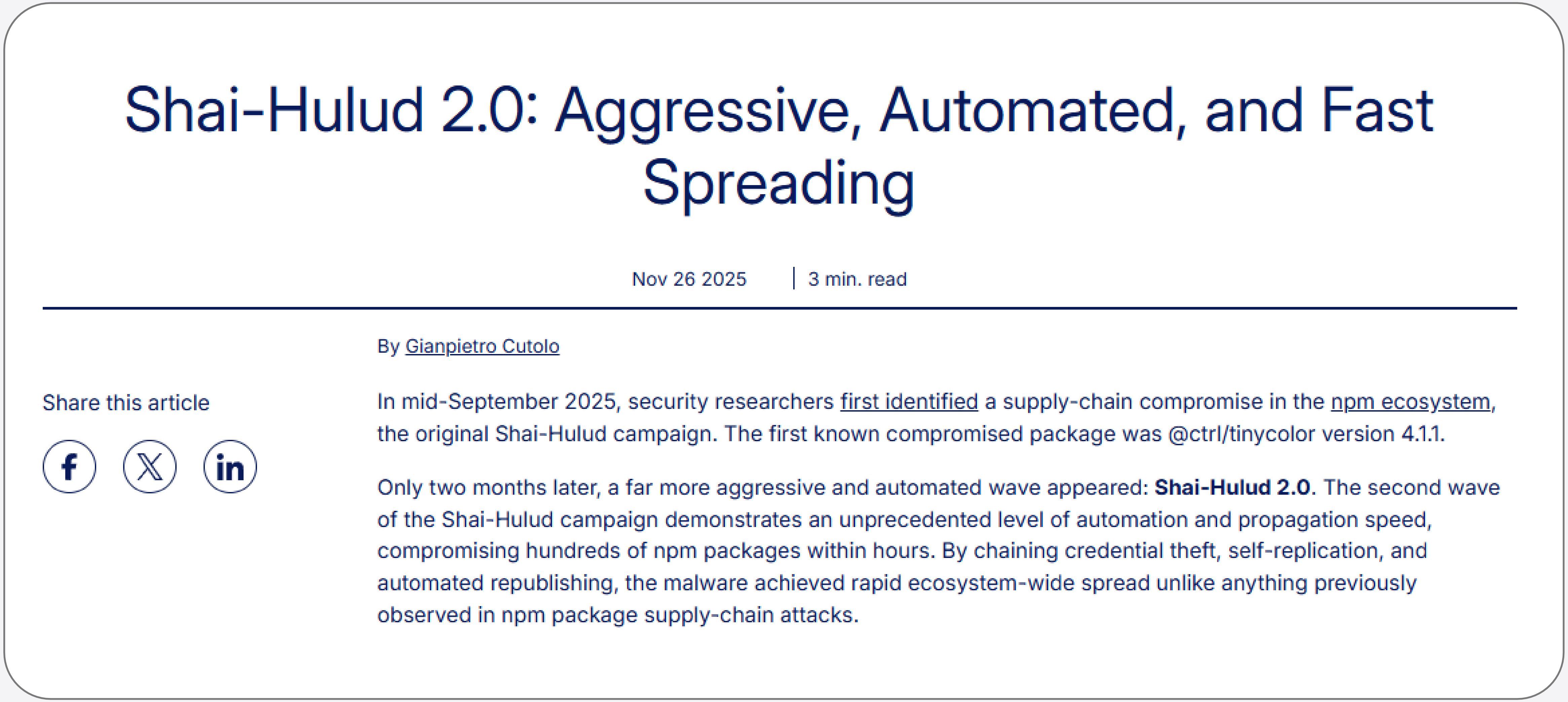
12 minute read time

As sure as long weekends arrive in the western world, so too does news of new software supply chain attacks. The easter bank holidays were no exception, with the discovery of a targeted attack against the popular XZ compression utility seen in many Linux distributions such as fedora, Debian to name a few.

The Sonatype team was alerted Friday (March 29, 2024) with the rest of the world as this attack was uncovered by a curious developer who noticed that their ssh login was taking 500ms instead of 100ms.

We think this is one of the more complicated benevolent stranger malware injections to date, and deserves amplification. This post is to discuss all the elements that have been discovered over the weekend and give our stance on this incident.

The practical end result is that the world now has another patching effort in front of them: to discover which systems are affected by the bad packages, and to upgrade to a known good version, which currently is understood to be anything below 5.6.0. The malicious code seems to have only been distributed in the operating system packages, and not present in the java-xz package. This may change as more research is performed.



Shai-Hulud 2.0: Aggressive, Automated, and Fast Spreading

Nov 26 2025 | 3 min. read

By Gianpietro Cutolo

Share this article

In mid-September 2025, security researchers first identified a supply-chain compromise in the npm ecosystem, the original Shai-Hulud campaign. The first known compromised package was @ctrl/tinycolor version 4.1.1.

Only two months later, a far more aggressive and automated wave appeared: **Shai-Hulud 2.0**. The second wave of the Shai-Hulud campaign demonstrates an unprecedented level of automation and propagation speed, compromising hundreds of npm packages within hours. By chaining credential theft, self-replication, and automated republishing, the malware achieved rapid ecosystem-wide spread unlike anything previously observed in npm package supply-chain attacks.

Vulnerabilities are ubiquitous

In 2025

npm recorded

838,778

releases associated with
CVSS 9.0+ vulnerabilities

1/5

PyPI releases was associated with a **CVSS 7.0+ vulnerability**

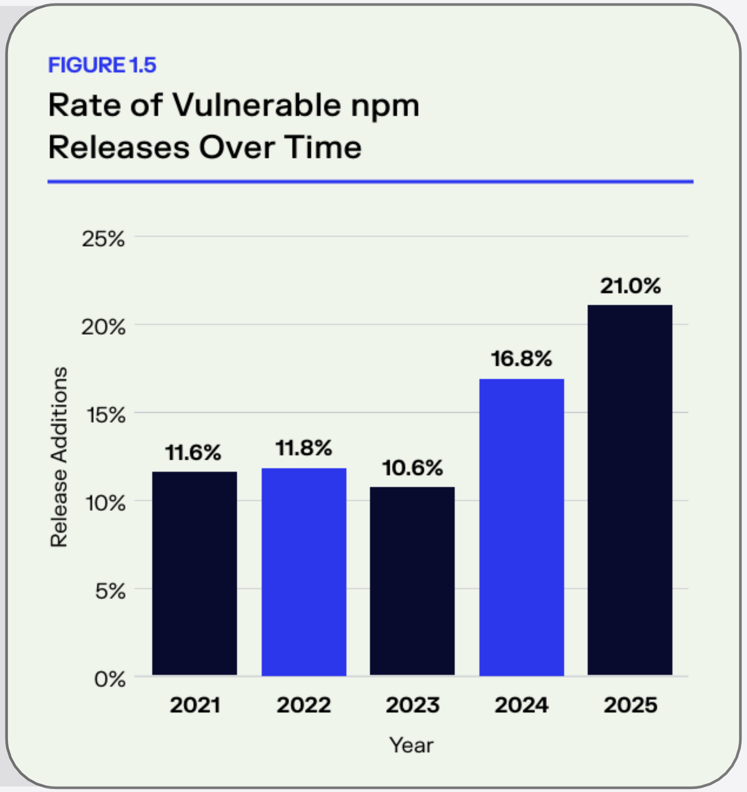


FIGURE 1.5

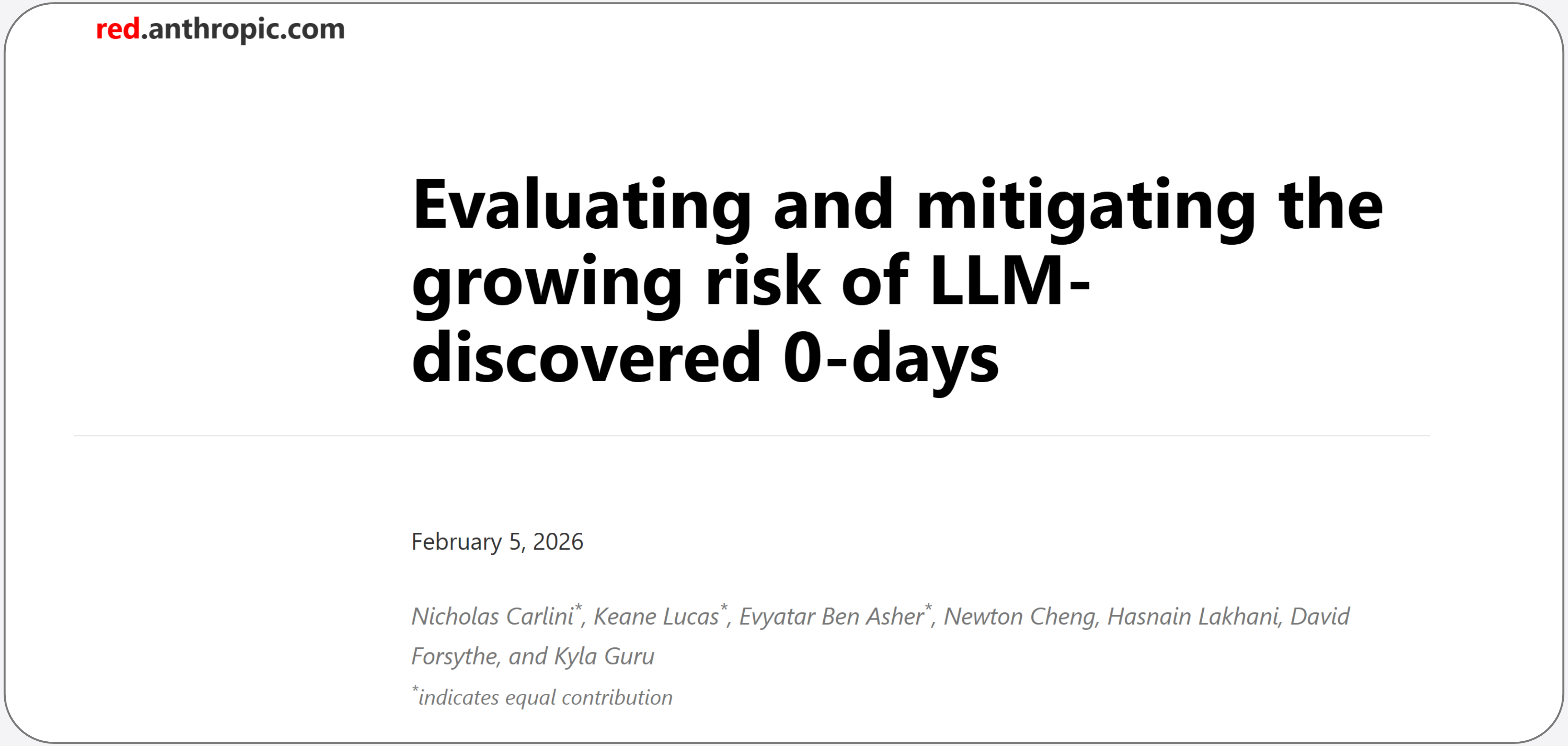
Rate of Vulnerable npm Releases Over Time

sonatype.com/state-of-the-software-supply-chain/2026/software-infrastructure-growth

Security risks of AI assisted coding

- AI suggests “popular” (historically common) versions, not secure ones
- AI generates manifests with outdated/vulnerable components
- Training data lags, so even after fixes exist, AI keeps suggesting vulnerable versions
- Without governance, AI increases component sprawl

The impending CVE deluge...



red.anthropic.com

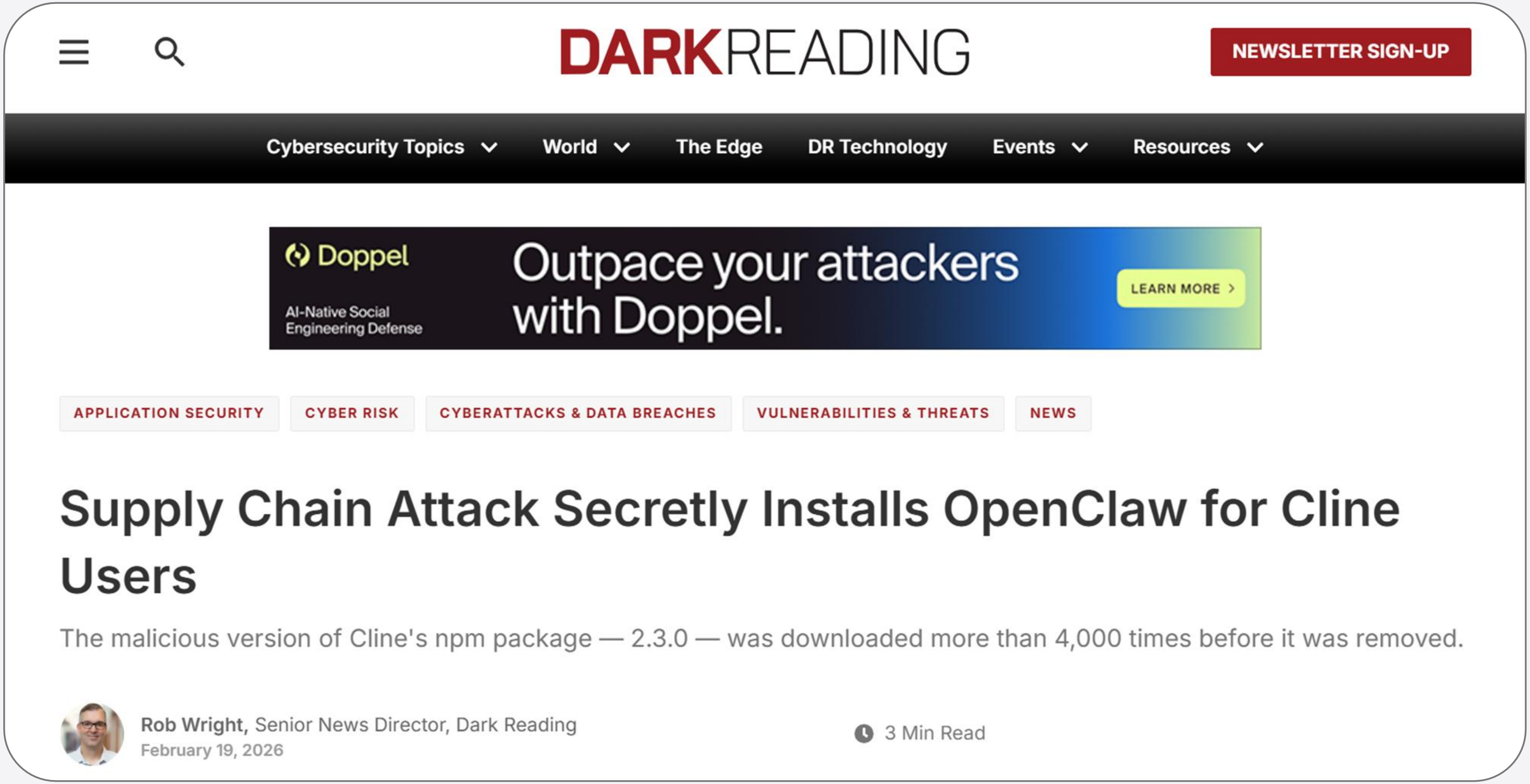
Evaluating and mitigating the growing risk of LLM-discovered 0-days

February 5, 2026

Nicholas Carlini[*], Keane Lucas[*], Evyatar Ben Asher[*], Newton Cheng, Hasnain Lakhani, David Forsythe, and Kyla Guru

[*] indicates equal contribution

Source: https://red.anthropic.com/2026/zero-days/



Supply Chain Attack Secretly Installs OpenClaw for Cline Users

The malicious version of Cline's npm package — 2.3.0 — was downloaded more than 4,000 times before it was removed.

Rob Wright, Senior News Director, Dark Reading
February 19, 2026

Regulatory reaction

May 12, 2021	US Executive Order 14028 signed.
July 12, 2021	US NTIA publishes SBOM "Minimum Elements."
Jan 16, 2023	NIS2 and DORA enter into force in the EU.
Dec 1, 2023	Australia's ASD ISM first edition released.
Oct 18, 2024	NIS2 compliance measures apply in the EU.
Dec 10, 2024	EU Cyber Resilience Act (CRA) entered into force.
Jan 17, 2025	DORA compliance measures apply in the EU.
Jul 25, 2025	CERT-In mandatory annual third-party cybersecurity audits in India.
Nov 12, 2025	UK CSRB introduced to parliament.

OpenSSF

OPEN SOURCE SECURITY FOUNDATION

Est. 2020

OpenSSF

OPEN SOURCE SECURITY FOUNDATION

Mission

The Open Source Security Foundation (OpenSSF) seeks to make it easier to **sustainably secure the development, maintenance, and consumption of the open source software (OSS) we all depend on**. This includes fostering collaboration, establishing best practices, and developing innovative solutions.

Vision

OSS is a digital public good and as an industry, we have an obligation to address the security concerns with the community. **We envision a future where OSS is universally trusted, secure, and reliable.** This collaborative vision enables individuals and organizations in a global ecosystem to confidently leverage the benefits and meaningfully contribute back to the OSS community.

OpenSSF

OPEN SOURCE SECURITY FOUNDATION

117 member organizations spanning **16 industries** (e.g., finance, cloud, AI, government, academia) and representation across **40+ countries**

Members Leading & Participation in Working Groups

Best Practices

- Microsoft
- intel.
- ERICSSON

ORBIT

- sonatype
- Red Hat
- KUSARI

AI/ML Security

- Microsoft
- DELL Technologies
- Google

Securing Critical Projects

- Google
- KUSARI
- CHAINGUARD

Securing Software Repositories

- Google
- GitHub
- IBM

Security Tooling

- intel.
- DELL Technologies
- ERICSSON

Supply Chain Integrity

- Microsoft
- IBM
- Red Hat

Vulnerability Disclosures

- Google
- GitHub
- CHAINGUARD

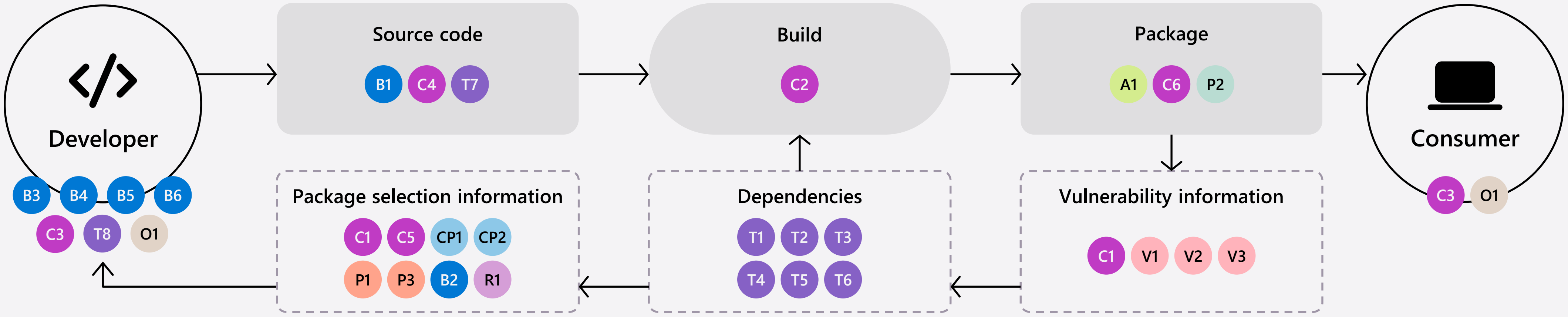
Global Cyber Policy

Belonging, Empowerment, Allyship, and Representation

Technical Advisory Council

Marketing Advisory Council

OpenSSF Technical Initiatives Landscape



Developer

B3 B4 B5 B6

C3 T8 O1

Source code

B1 C4 T7

Build

C2

Package

A1 C6 P2

Consumer

C3 O1

Package selection information

C1 C5 CP1 CP2

P1 P3 B2 R1

Dependencies

T1 T2 T3

T4 T5 T6

Vulnerability information

C1 V1 V2 V3

A **AI/ML Security**

1. Model Signing SIG & Project

B **Best Practices**

1. OpenSSF Best Practices Badge project
2. OpenSSF Scorecard project
3. Education SIG
4. Memory Safety SIG
5. C/C++ Compiler Options SIG
6. Python Hardening SIG

CP **Securing Critical Projects**

1. Criticality_score project
2. Package Analysis project

C **Supply Chain Integrity**

1. Security Insights project
2. SLSA project
3. S2C2F project
4. Gittuf project
5. GUAC project
6. Zarf project M

BEAR (Belonging, Empowerment, Allyship, and Representation)

R **Securing Software Repositories**

1. **RSTUF Project**

P **Projects**

1. Alpha & Omega project
2. Sigstore
3. Core Toolchain Infrastructure (CTI)

T **Security Tooling**

1. SBOM Everywhere SIG
2. OSS Fuzzing SIG
3. SBOMit project
4. Protobom project
5. bomctl project
6. Fuzz Introspector project
7. Minder project
8. OpenBao project

V **Vulnerability Disclosures**

1. CVD Guides SIGs
2. OSV Schema project
3. OpenVEX SIG
   OpenVEX Project

O **ORBIT (Open Resources for Baselines, Interoperability, and Tooling)**

1. OSPS Baseline project

Global Cyber Policy

DevRel Community

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

Supply-chain Levels for Software Artifacts (SLSA)

Safeguarding artifact integrity across any software supply chain

SLSA: Source Track - Levels of Assurance

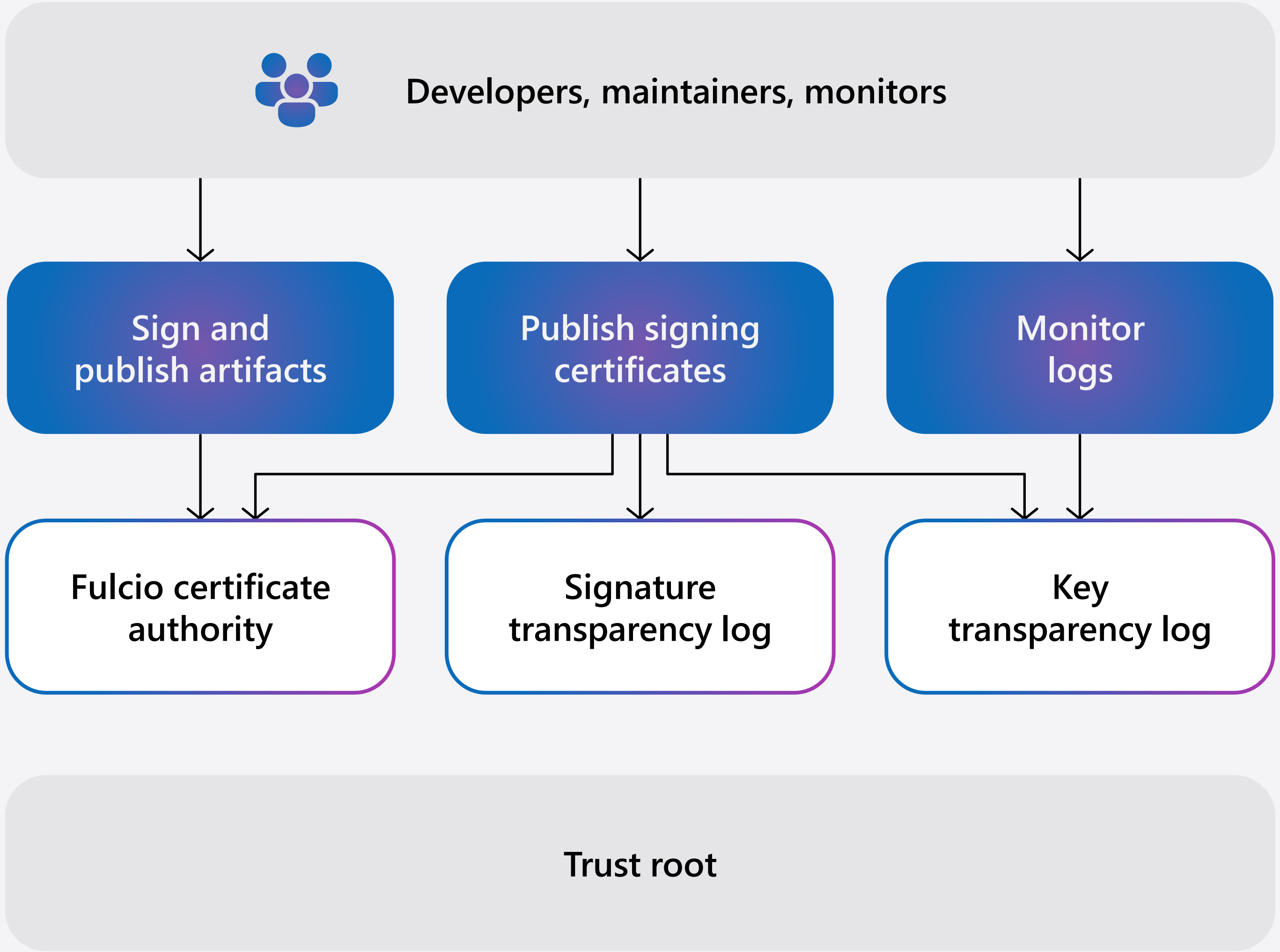
Track/Level	Requirements	Focus
Source L1	Use a version control system	Generation of discrete Source Revisions for precise consumption
Source L2	Preserve Change History and generate Source Provenance	Reliable history through enforced controls and evidence
Source L3	Enforce organizational technical controls	Consumer knowledge of guaranteed technical controls
Source L4	Require code review	Improved code quality and resistance to insider threats

SLSA: Build Track - Levels of Assurance

Track/Level	Requirements	Focus
Build L0	(none)	(n/a)
Build L1	Provenance showing how the package was built	Mistakes, documentation
Build L2	Signed provenance, generated by a hosted build platform	Tampering after the build
Build L3	Hardened build platform	Tampering during the build

Supply Chain Transparency Standards/Frameworks

Sigstore



Developers, maintainers, monitors

Sign and publish artifacts

Publish signing certificates

Monitor logs

Fulcio certificate authority

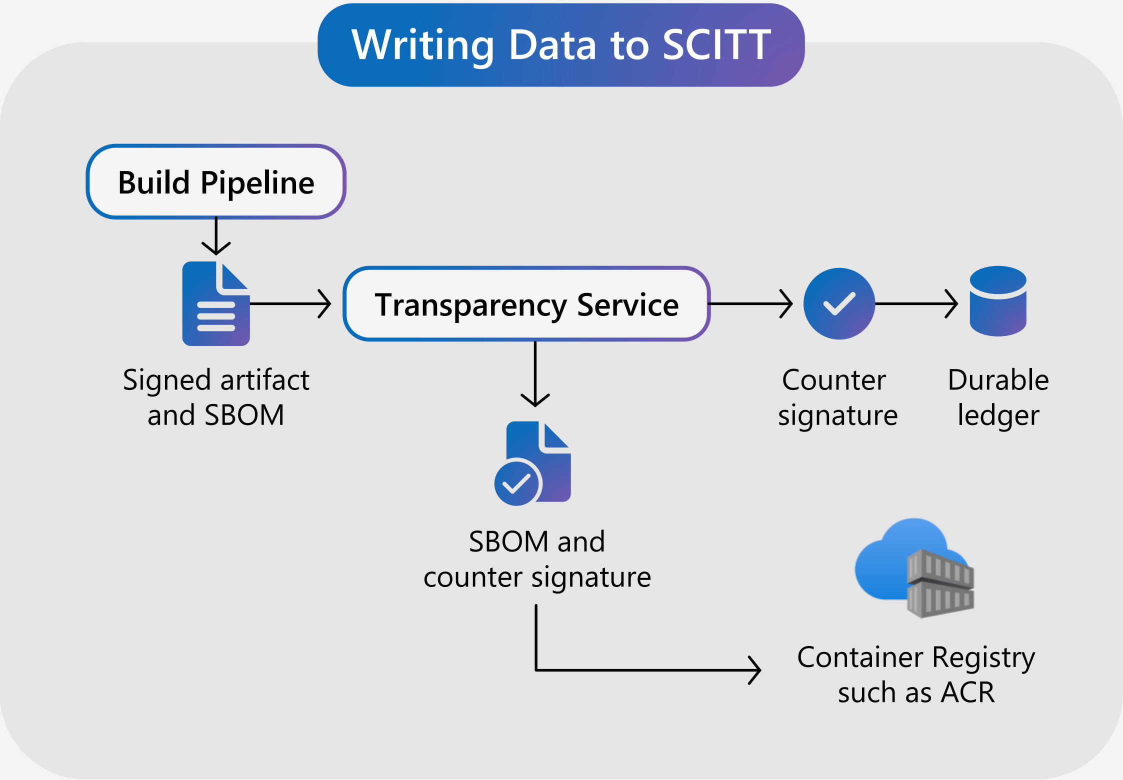
Signature transparency log

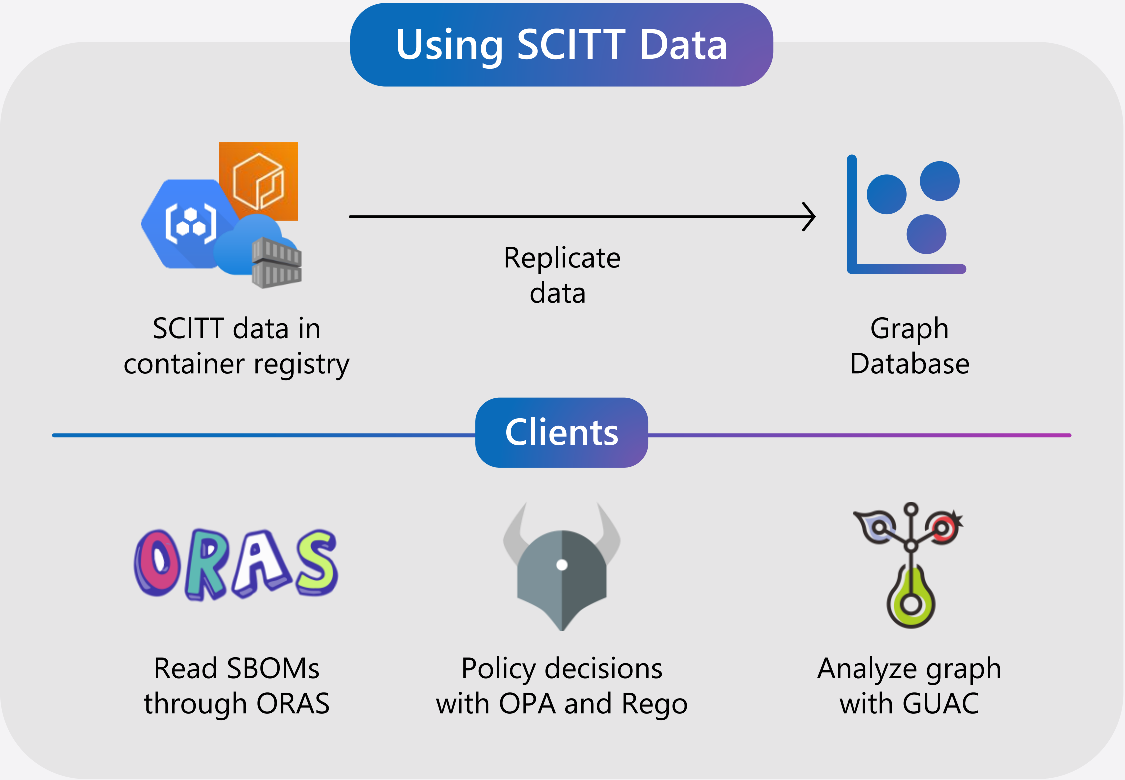
Key transparency log

Trust root

Supply Chain Integrity Transparency and Trust (SCITT)

An IETF project to provide a generic, interoperable, and scalable architecture to enable transparency across any supply chain with minimum adoption barriers





Implementers of SCITT can use their preferred durable ledger, container registry, and graph databases as part of their implementation

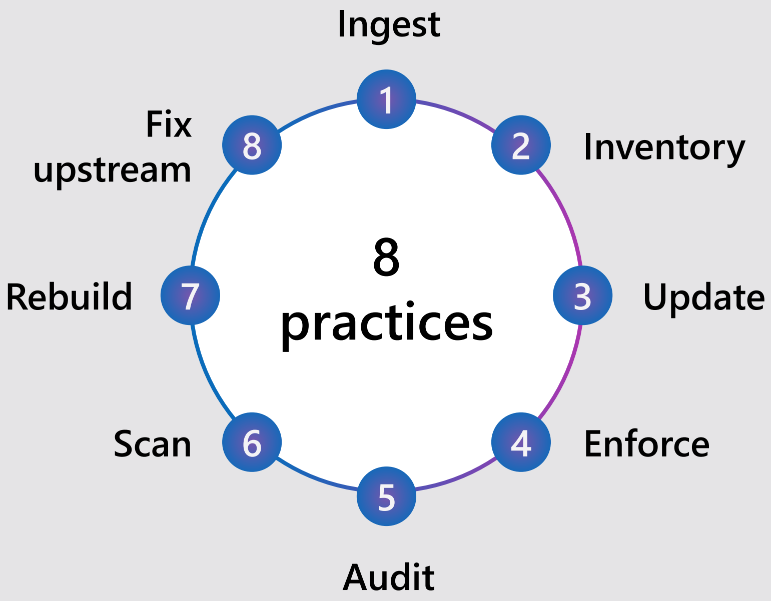
OpenSSF
OPEN SOURCE SECURITY FOUNDATION

Secure Supply Chain Consumption Framework (S2C2F)

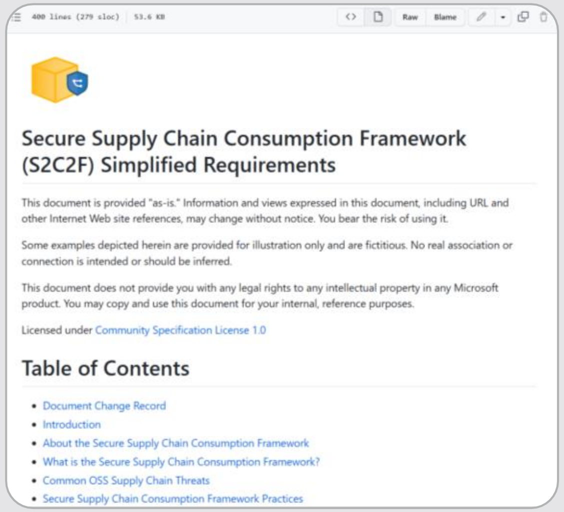
The S2C2F guide outlines and defines how to securely consume OSS dependencies into the developer’s workflow.

S2C2F

Microsoft defined framework that is contributed to the OpenSSF in 2022



Each practice has specific requirements created using a threat-based risk-reduction approach toward secure consumption



Secure Supply Chain Consumption Framework (S2C2F) Simplified Requirements

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

Licensed under Community Specification License 1.0

Table of Contents

High-level solution-agnostic set of practices

Detailed list of requirements for each practice

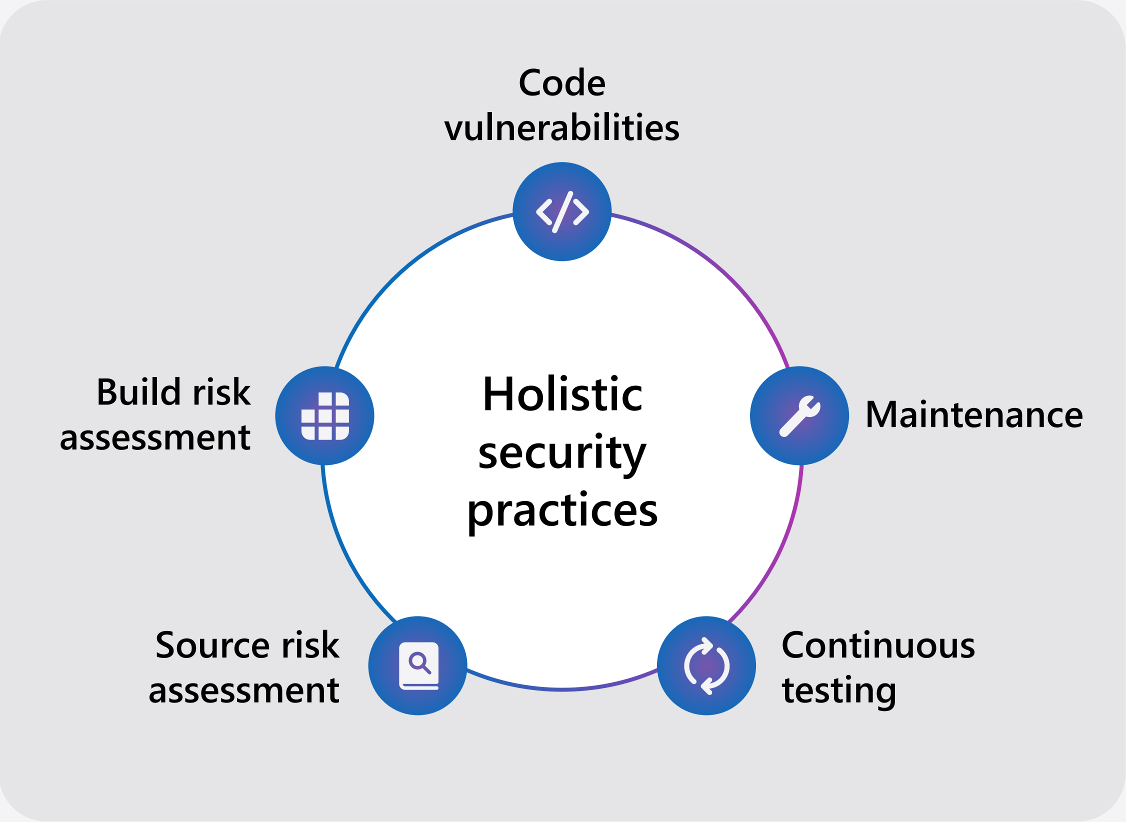
Real-world supply chain threats specific to OSS, and how our Framework requirements mitigates them

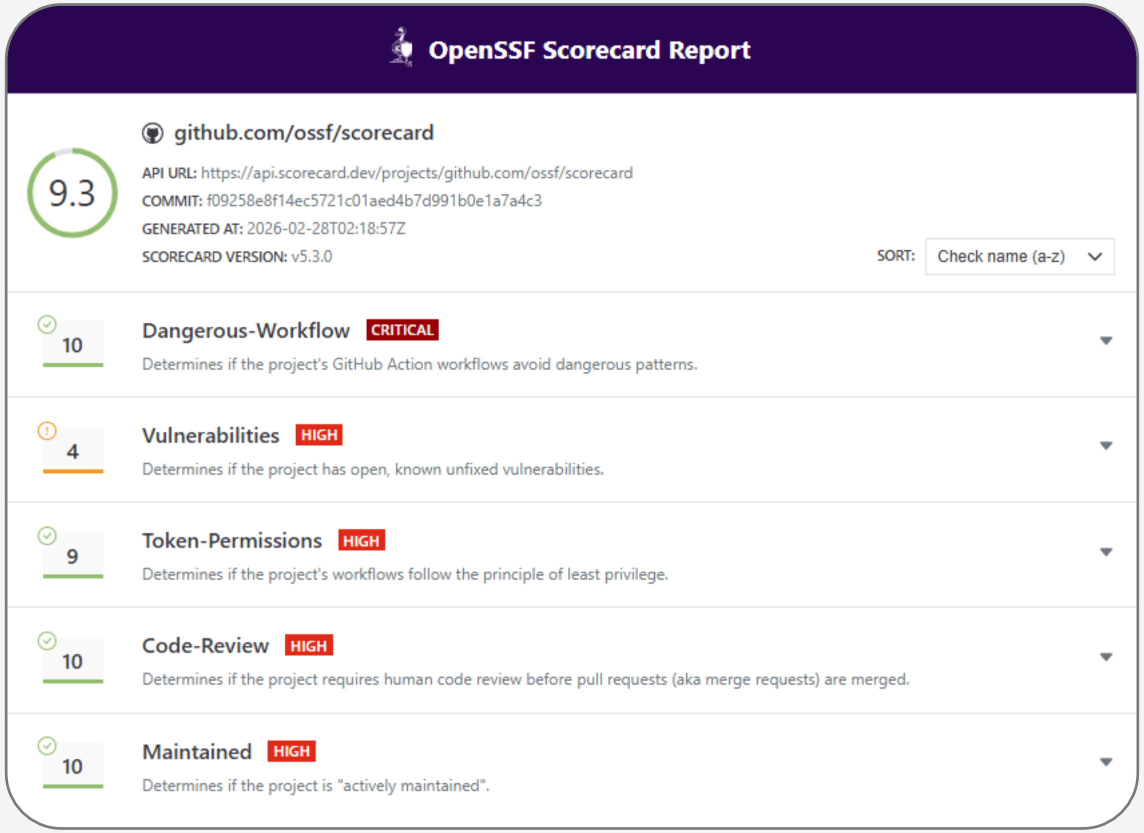
OpenSSF
OPEN SOURCE SECURITY FOUNDATION

OpenSSF Scorecard

Quickly assess open source projects for risky practices

Automated checks with weighted scoring





OpenSSF Scorecard Report

9.3

github.com/ossf/scorecard

API URL: https://api.scorecard.dev/projects/github.com/ossf/scorecard
COMMIT: f09258e8f14ec5721c01aed4b7d991b0e1a7a4c3
GENERATED AT: 2026-02-28T02:18:57Z
SCORECARD VERSION: v5.3.0

SORT: Check name (a-z)

✓ 10 Dangerous-Workflow CRITICAL
Determines if the project's GitHub Action workflows avoid dangerous patterns.

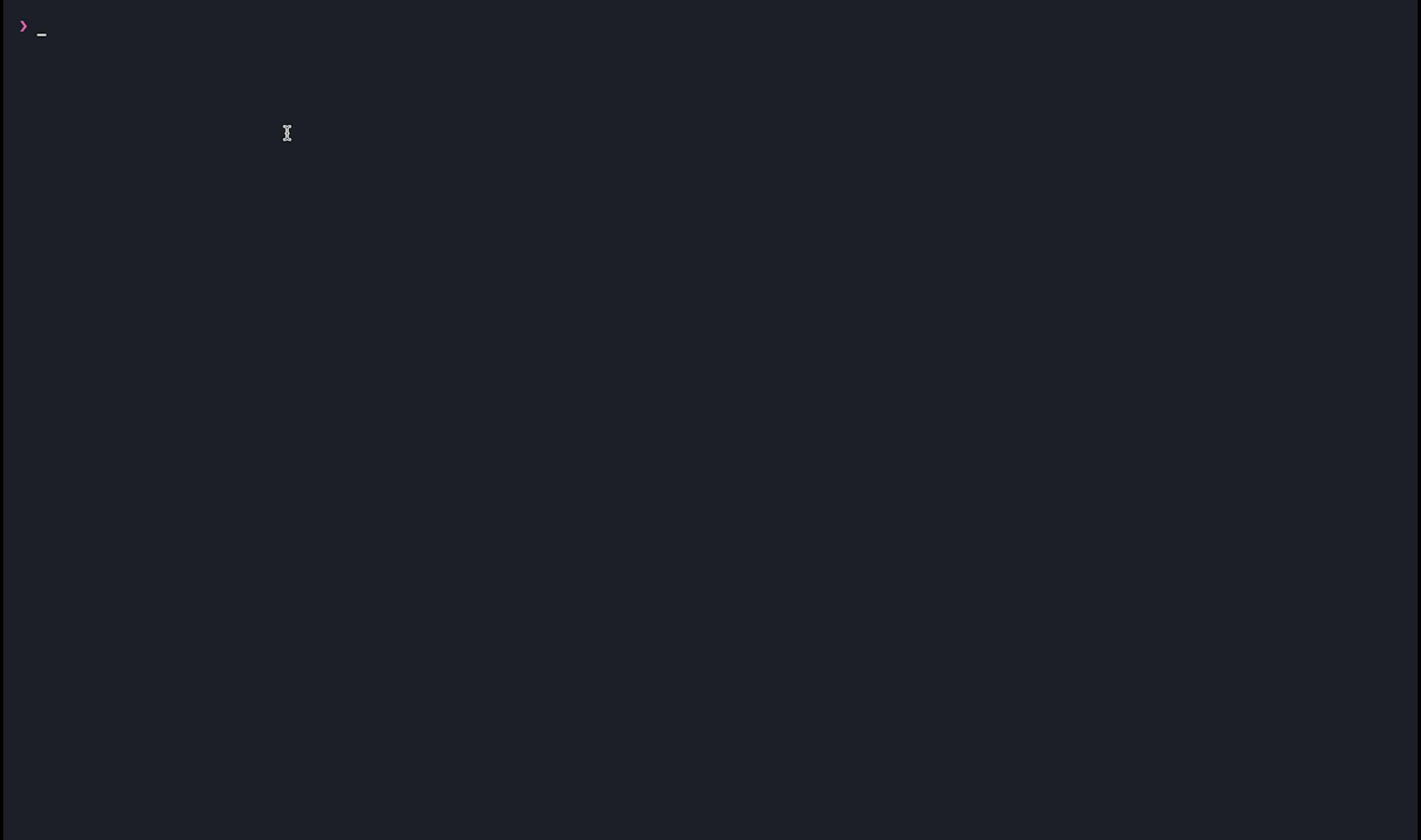
! 4 Vulnerabilities HIGH
Determines if the project has open, known unfixed vulnerabilities.

✓ 9 Token-Permissions HIGH
Determines if the project's workflows follow the principle of least privilege.

✓ 10 Code-Review HIGH
Determines if the project requires human code review before pull requests (aka merge requests) are merged.

✓ 10 Maintained HIGH
Determines if the project is "actively maintained".

securityscorecards.dev/viewer/?uri=github.com%2Fossf%2Fscorecard



```
❯
```

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

Open Source Project Security Baseline

Structured security requirements aligned with international frameworks, standards, and regulations

Baseline levels

1

20

“Universal security floor” for all open source - great for single maintainer or early maturity projects

Are you a Foundation? The level 1 baseline should be your first set of criteria for maturing projects (or even accepting projects)

2

18

Good for projects with 2 - 6 maintainers and maturing

3

9

Security flex - good for highly mature projects that consider security a core competency

Are you in a Foundation with project resources? You should strive for this one

https://baseline.openssf.org
https://github.com/ossf/security-baseline

Library of practices aligned to requirements

THE BASE

NIST National Institute of Standards and Technology
U.S. Department of Commerce

SSDF

CSF

800-161/800-53

CISA Software Acquisition Guide

Forthcoming

European Commission

Cyber Resilience Act

DORA Forthcoming

National Cyber Security Centre
a part of GCHQ

Software Security Code of Practice

Forthcoming

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

BP Badges

Scorecard

Minder

SLSA

OpenSSF tooling

PCI Security Standards Council

SAMM

OPENCRE

OPENCHAIN

Proactive Software Supply Chain Risk Management (P-SSCRM) Framework

Alpha Omega

Secure the most critical open source software projects and ecosystems

α → Leverage

Secures the most critical open source projects (via OpenSSF Criticality Score and Harvard Census)

Funds threat modeling, testing, audits, remediation, and Scorecard/Best Practices alignment

Ω → Scale

Identifies and fixes critical vulnerabilities across 10,000+ widely used OSS projects

Combines automation, security analysts, and coordinated disclosure

Alpha Omega

Secure the most critical open source software projects and ecosystems

In 2025,
$5.8 million
funded 14 critical
open-source
projects.

- → Rust's Trusted Publishing launch and CVE authority designation
- → Apache's Trusted Release pipeline pilot
- → Security upgrades across Python, Node.js, Ruby, Eclipse Foundation, and FreeBSD
- → Audits of numerous open source projects

Choose your role, then do one concrete thing

If you maintain open source

Publish *what you ship* (signed releases, provenance)

Be explicit about security posture (even if it's "best effort")

Ask for help — funding, audits, automation

If you build on open source

Know what you're pulling in (inventory your dependencies)

Don't ignore known vulnerabilities "because upstream"

Prefer projects that show security hygiene, and tell them why

If your business depends on open source

Fund it: money, engineers, or infrastructure

Support maintainers **before** the next incident

Treat open source as critical infrastructure, not free input

The Digital Window: Computer Graphics Today And Tomorrow

COMPUTE!

$2.95
May
1986
Issue 72
Vol. 8, No. 5
$3.75 Canada
02193
ISSN 0194-357X

The Leading Magazine Of Home, Educational, And Recreational Computing

Two New Monthly Columns
Insight:ST And AmigaView

Managing Files From Atari ST BASIC: A Major Book Excerpt

Better Branching In Applesoft
Computed GOTO And GOSUB

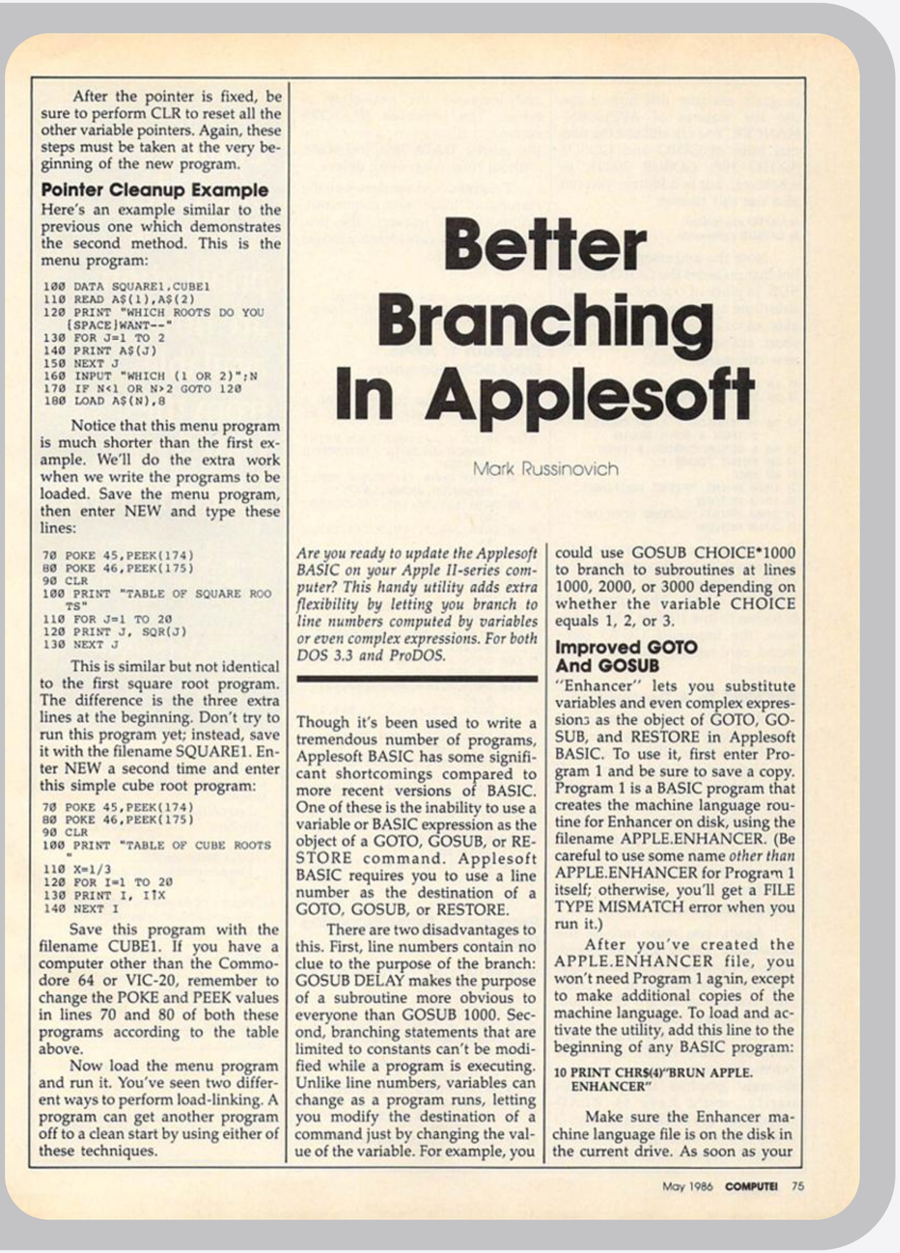
64 Autobooter
Make Disk Programs Run Automatically

Atari DEBUT:
Add Commands To BASIC For Easy Debugging

Hickory, Dickory, Dock
An Educational Game For Commodore 64 And 128, Atari, Apple, Atari ST, Amiga, And IBM PC/PCjr

MODified Shapes For IBM

Amiga Puzzle
Learn New Amiga BASIC Programming Techniques



After the pointer is fixed, be sure to perform CLR to reset all the other variable pointers. Again, these steps must be taken at the very beginning of the new program.

Pointer Cleanup Example

Here's an example similar to the previous one which demonstrates the second method. This is the menu program:

```
100 DATA SQUARE1,CUBE1
110 READ A$(1),A$(2)
120 PRINT "WHICH ROOTS DO YOU
   {SPACE}WANT--"
130 FOR J=1 TO 2
140 PRINT A$(J)
150 NEXT J
160 INPUT "WHICH (1 OR 2)";N
170 IF N<1 OR N>2 GOTO 120
180 LOAD A$(N),8
```

Notice that this menu program is much shorter than the first example. We'll do the extra work when we write the programs to be loaded. Save the menu program, then enter NEW and type these lines:

```
70 POKE 45,PEEK(174)
80 POKE 46,PEEK(175)
90 CLR
100 PRINT "TABLE OF SQUARE ROO
   TS"
110 FOR J=1 TO 20
120 PRINT J, SQR(J)
130 NEXT J
```

This is similar but not identical to the first square root program. The difference is the three extra lines at the beginning. Don't try to run this program yet; instead, save it with the filename SQUARE1. Enter NEW a second time and enter this simple cube root program:

```
70 POKE 45,PEEK(174)
80 POKE 46,PEEK(175)
90 CLR
100 PRINT "TABLE OF CUBE ROOTS
   "
110 X=1/3
120 FOR I=1 TO 20
130 PRINT I, I↑X
140 NEXT I
```

Save this program with the filename CUBE1. If you have a computer other than the Commodore 64 or VIC-20, remember to change the POKE and PEEK values in lines 70 and 80 of both these programs according to the table above.

Now load the menu program and run it. You've seen two different ways to perform load-linking. A program can get another program off to a clean start by using either of these techniques.

Better Branching In Applesoft

Mark Russinovich

Are you ready to update the Applesoft BASIC on your Apple II-series computer? This handy utility adds extra flexibility by letting you branch to line numbers computed by variables or even complex expressions. For both DOS 3.3 and ProDOS.

Though it's been used to write a tremendous number of programs, Applesoft BASIC has some significant shortcomings compared to more recent versions of BASIC. One of these is the inability to use a variable or BASIC expression as the object of a GOTO, GOSUB, or RESTORE command. Applesoft BASIC requires you to use a line number as the destination of a GOTO, GOSUB, or RESTORE.

There are two disadvantages to this. First, line numbers contain no clue to the purpose of the branch: GOSUB DELAY makes the purpose of a subroutine more obvious to everyone than GOSUB 1000. Second, branching statements that are limited to constants can't be modified while a program is executing. Unlike line numbers, variables can change as a program runs, letting you modify the destination of a command just by changing the value of the variable. For example, you could use GOSUB CHOICE*1000 to branch to subroutines at lines 1000, 2000, or 3000 depending on whether the variable CHOICE equals 1, 2, or 3.

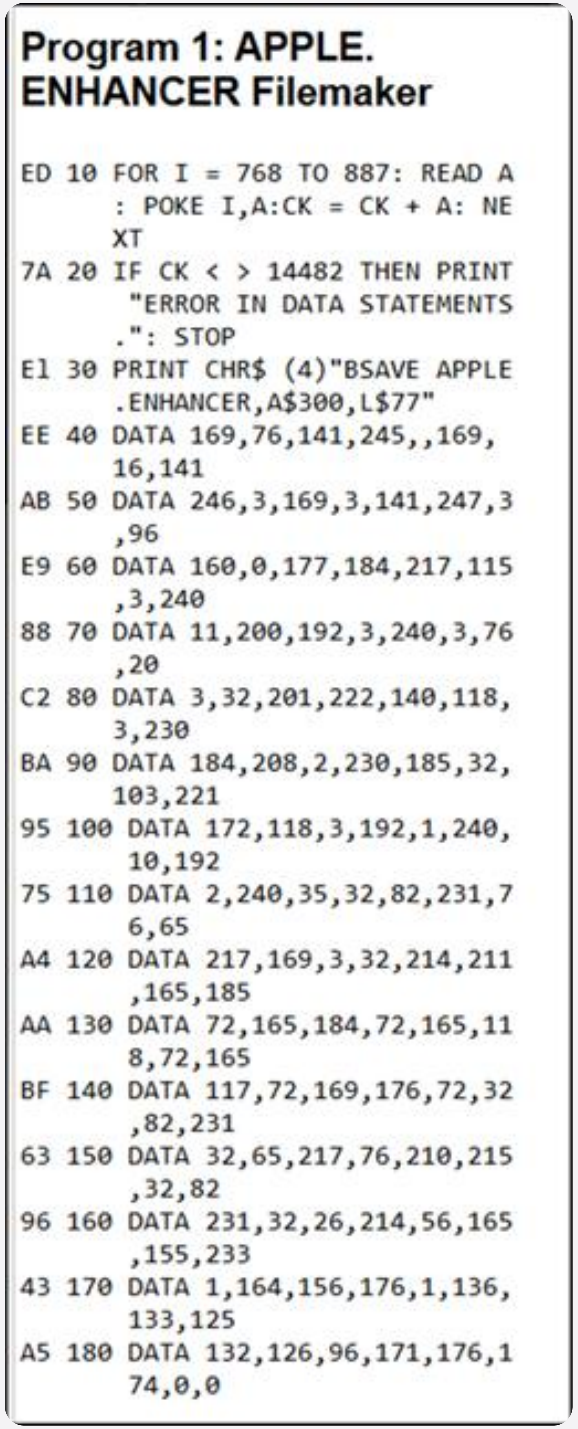
Improved GOTO And GOSUB

"Enhancer" lets you substitute variables and even complex expressions as the object of GOTO, GOSUB, and RESTORE in Applesoft BASIC. To use it, first enter Program 1 and be sure to save a copy. Program 1 is a BASIC program that creates the machine language routine for Enhancer on disk, using the filename APPLE.ENHANCER. (Be careful to use some name *other than* APPLE.ENHANCER for Program 1 itself; otherwise, you'll get a FILE TYPE MISMATCH error when you run it.)

After you've created the APPLE.ENHANCER file, you won't need Program 1 again, except to make additional copies of the machine language. To load and activate the utility, add this line to the beginning of any BASIC program:

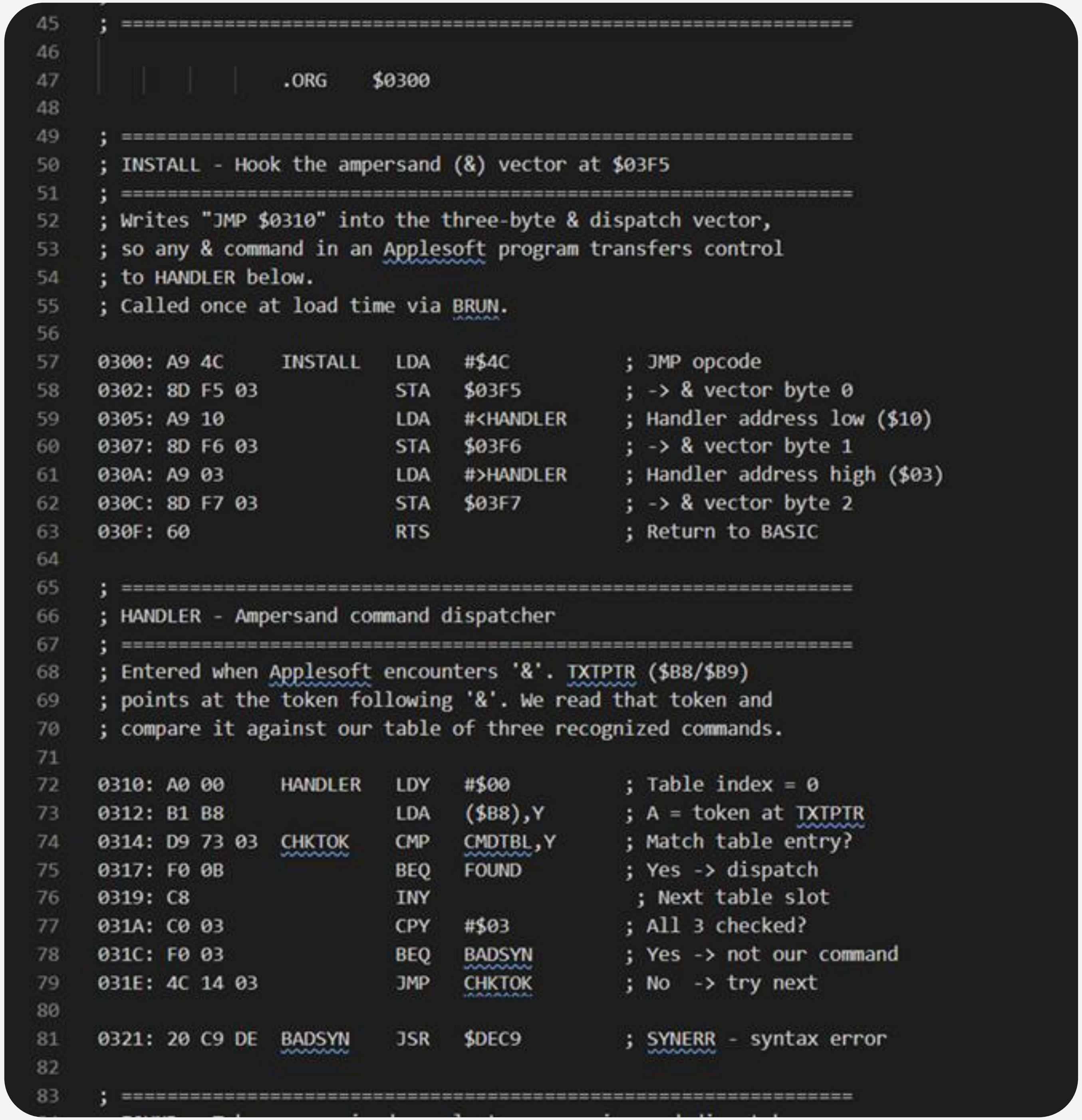
```
10 PRINT CHR$(4)"BRUN APPLE.
   ENHANCER"
```

Make sure the Enhancer machine language file is on the disk in the current drive. As soon as your

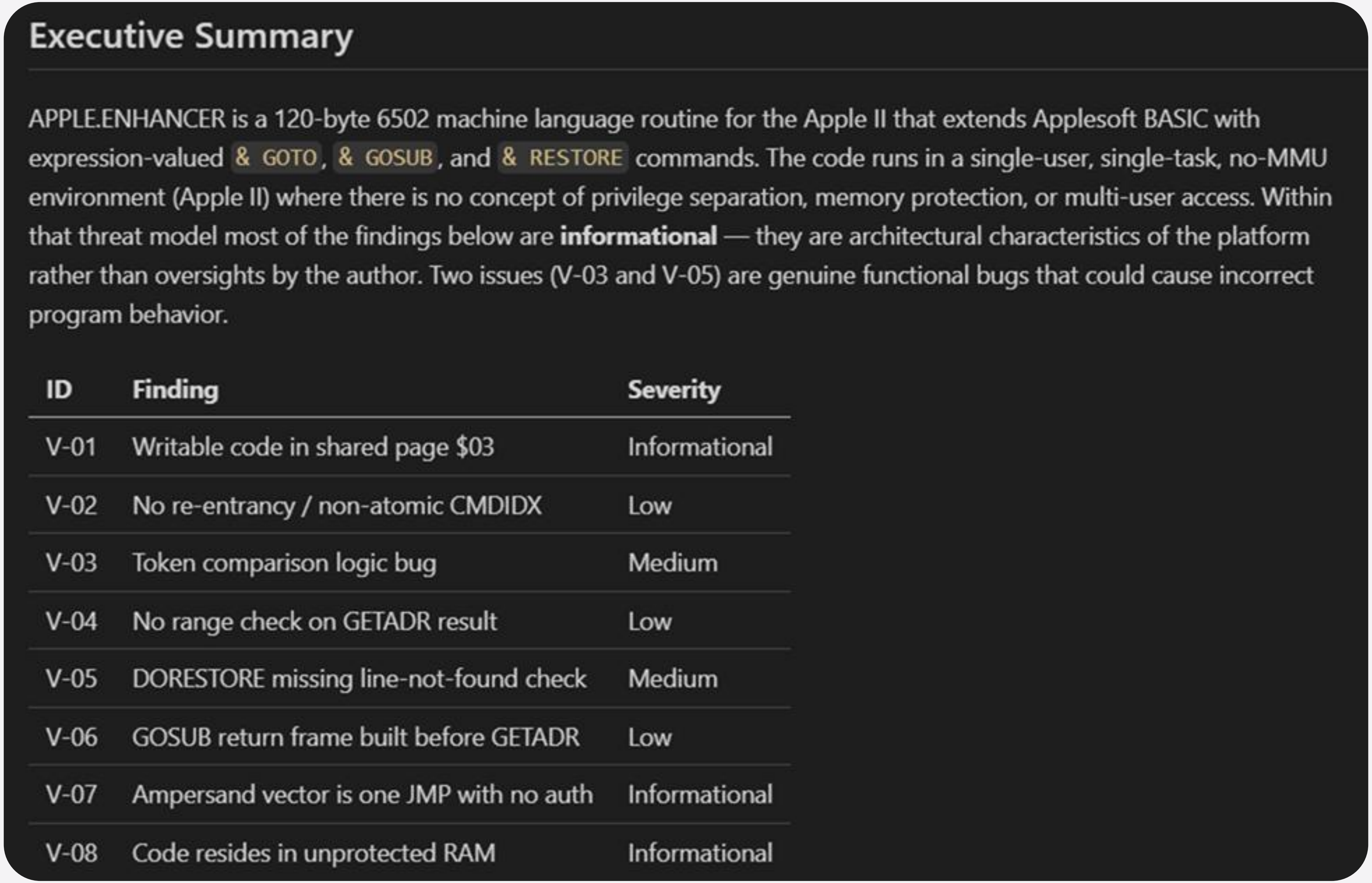


Program 1: APPLE. ENHANCER Filemaker

```
ED 10 FOR I = 768 TO 887: READ A
      : POKE I,A:CK = CK + A: NE
      XT
7A 20 IF CK < > 14482 THEN PRINT
       "ERROR IN DATA STATEMENTS
      .": STOP
E1 30 PRINT CHR$ (4)"BSAVE APPLE
      .ENHANCER,A$300,L$77"
EE 40 DATA 169,76,141,245,,169,
      16,141
AB 50 DATA 246,3,169,3,141,247,3
      ,96
E9 60 DATA 160,0,177,184,217,115
      ,3,240
88 70 DATA 11,200,192,3,240,3,76
      ,20
C2 80 DATA 3,32,201,222,140,118,
      3,230
BA 90 DATA 184,208,2,230,185,32,
      103,221
95 100 DATA 172,118,3,192,1,240,
       10,192
75 110 DATA 2,240,35,32,82,231,7
       6,65
A4 120 DATA 217,169,3,32,214,211
       ,165,185
AA 130 DATA 72,165,184,72,165,11
       8,72,165
BF 140 DATA 117,72,169,176,72,32
       ,82,231
63 150 DATA 32,65,217,76,210,215
       ,32,82
96 160 DATA 231,32,26,214,56,165
       ,155,233
43 170 DATA 1,164,156,176,1,136,
       133,125
A5 180 DATA 132,126,96,171,176,1
       74,0,0
```



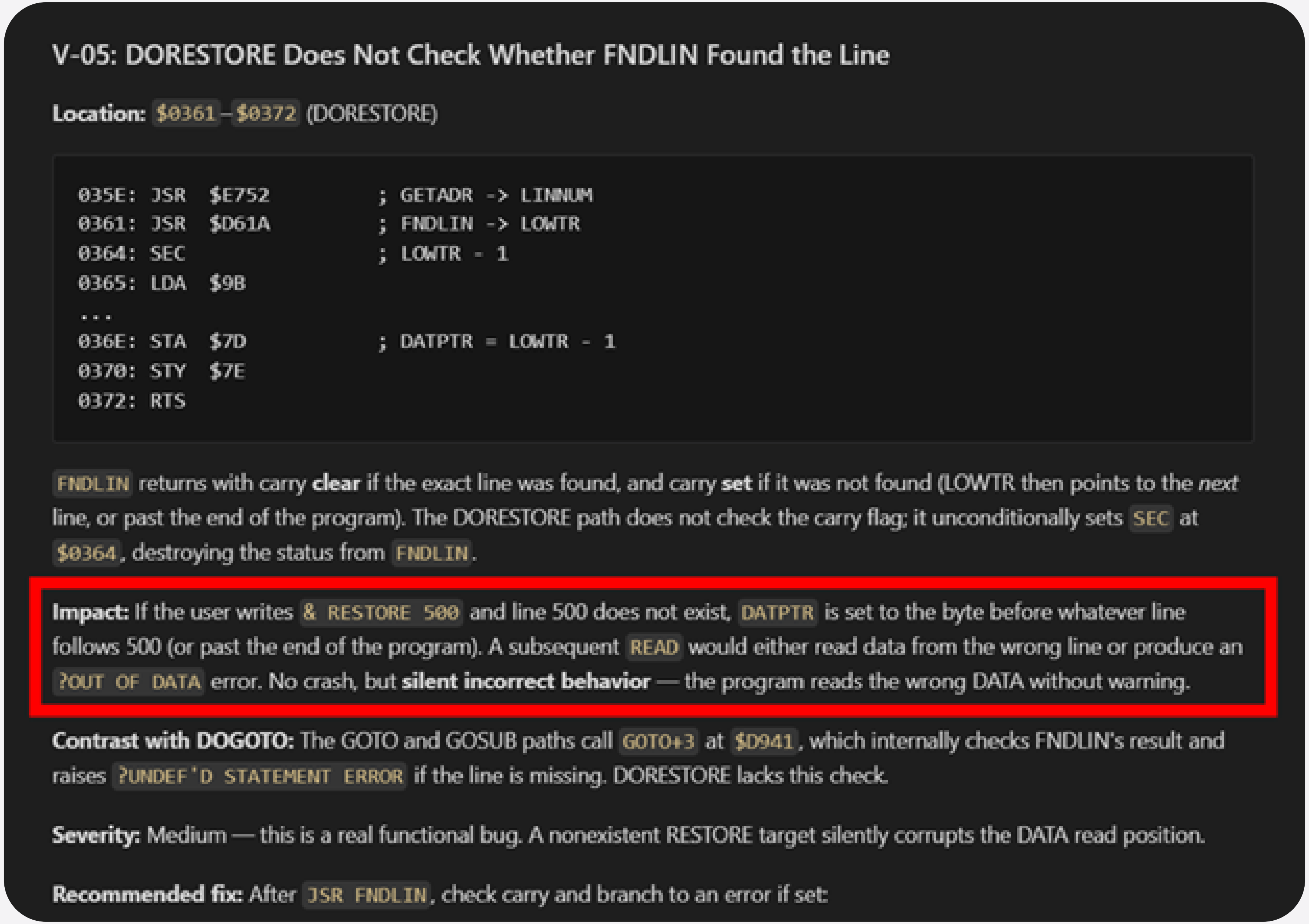
```
; ================================================================

                .ORG    $0300

; ================================================================
; INSTALL - Hook the ampersand (&) vector at $03F5
; ================================================================
; Writes "JMP $0310" into the three-byte & dispatch vector,
; so any & command in an Applesoft program transfers control
; to HANDLER below.
; Called once at load time via BRUN.

0300: A9 4C     INSTALL   LDA   #$4C          ; JMP opcode
0302: 8D F5 03            STA   $03F5         ; -> & vector byte 0
0305: A9 10               LDA   #<HANDLER     ; Handler address low ($10)
0307: 8D F6 03            STA   $03F6         ; -> & vector byte 1
030A: A9 03               LDA   #>HANDLER     ; Handler address high ($03)
030C: 8D F7 03            STA   $03F7         ; -> & vector byte 2
030F: 60                  RTS                 ; Return to BASIC

; ================================================================
; HANDLER - Ampersand command dispatcher
; ================================================================
; Entered when Applesoft encounters '&'. TXTPTR ($B8/$B9)
; points at the token following '&'. We read that token and
; compare it against our table of three recognized commands.

0310: A0 00     HANDLER   LDY   #$00          ; Table index = 0
0312: B1 B8               LDA   ($B8),Y       ; A = token at TXTPTR
0314: D9 73 03  CHKTOK    CMP   CMDTBL,Y      ; Match table entry?
0317: F0 0B               BEQ   FOUND         ; Yes -> dispatch
0319: C8                  INY                  ; Next table slot
031A: C0 03               CPY   #$03          ; All 3 checked?
031C: F0 03               BEQ   BADSYN        ; Yes -> not our command
031E: 4C 14 03            JMP   CHKTOK        ; No  -> try next

0321: 20 C9 DE  BADSYN    JSR   $DEC9         ; SYNERR - syntax error

; ================================================================
```



Executive Summary

APPLE.ENHANCER is a 120-byte 6502 machine language routine for the Apple II that extends Applesoft BASIC with expression-valued `& GOTO`, `& GOSUB`, and `& RESTORE` commands. The code runs in a single-user, single-task, no-MMU environment (Apple II) where there is no concept of privilege separation, memory protection, or multi-user access. Within that threat model most of the findings below are **informational** — they are architectural characteristics of the platform rather than oversights by the author. Two issues (V-03 and V-05) are genuine functional bugs that could cause incorrect program behavior.

ID	Finding	Severity
V-01	Writable code in shared page $03	Informational
V-02	No re-entrancy / non-atomic CMDIDX	Low
V-03	Token comparison logic bug	Medium
V-04	No range check on GETADR result	Low
V-05	DORESTORE missing line-not-found check	Medium
V-06	GOSUB return frame built before GETADR	Low
V-07	Ampersand vector is one JMP with no auth	Informational
V-08	Code resides in unprotected RAM	Informational



V-05: DORESTORE Does Not Check Whether FNDLIN Found the Line

Location: `$0361`–`$0372` (DORESTORE)

```
035E: JSR  $E752         ; GETADR -> LINNUM
0361: JSR  $D61A         ; FNDLIN -> LOWTR
0364: SEC                ; LOWTR - 1
0365: LDA  $9B
...
036E: STA  $7D           ; DATPTR = LOWTR - 1
0370: STY  $7E
0372: RTS
```

`FNDLIN` returns with carry **clear** if the exact line was found, and carry **set** if it was not found (LOWTR then points to the *next* line, or past the end of the program). The DORESTORE path does not check the carry flag; it unconditionally sets `SEC` at `$0364`, destroying the status from `FNDLIN`.

Impact: If the user writes `& RESTORE 500` and line 500 does not exist, `DATPTR` is set to the byte before whatever line follows 500 (or past the end of the program). A subsequent `READ` would either read data from the wrong line or produce an `?OUT OF DATA` error. No crash, but **silent incorrect behavior** — the program reads the wrong DATA without warning.

Contrast with DOGOTO: The GOTO and GOSUB paths call `GOTO+3` at `$D941`, which internally checks FNDLIN's result and raises `?UNDEF'D STATEMENT ERROR` if the line is missing. DORESTORE lacks this check.

Severity: Medium — this is a real functional bug. A nonexistent RESTORE target silently corrupts the DATA read position.

Recommended fix: After `JSR FNDLIN`, check carry and branch to an error if set:

Secure your Code, Repos, Pipelines

Join a OpenSSF Working Group:

Contribute to ongoing security initiatives

Openssf.org/getinvolved/

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

2025
Annual
Report

openssf.org