Frog Proof Security

How to Confidently Ignore 88% of Your Critical Vulnerabilities

Mike Holland
Senior Solutions Engineer, JFrog

Binaries are the SINGLE SOURCE for All Your Software

New Developer Tools

are Constantly and Rapidly Introduced

GitHub Copilot

Sourcegraph Cody

Windsurf

Amazon Q

Claude Code

JFrog fly

Cline

Gemini

Goose

Qodo

Codex

Devin

Augment

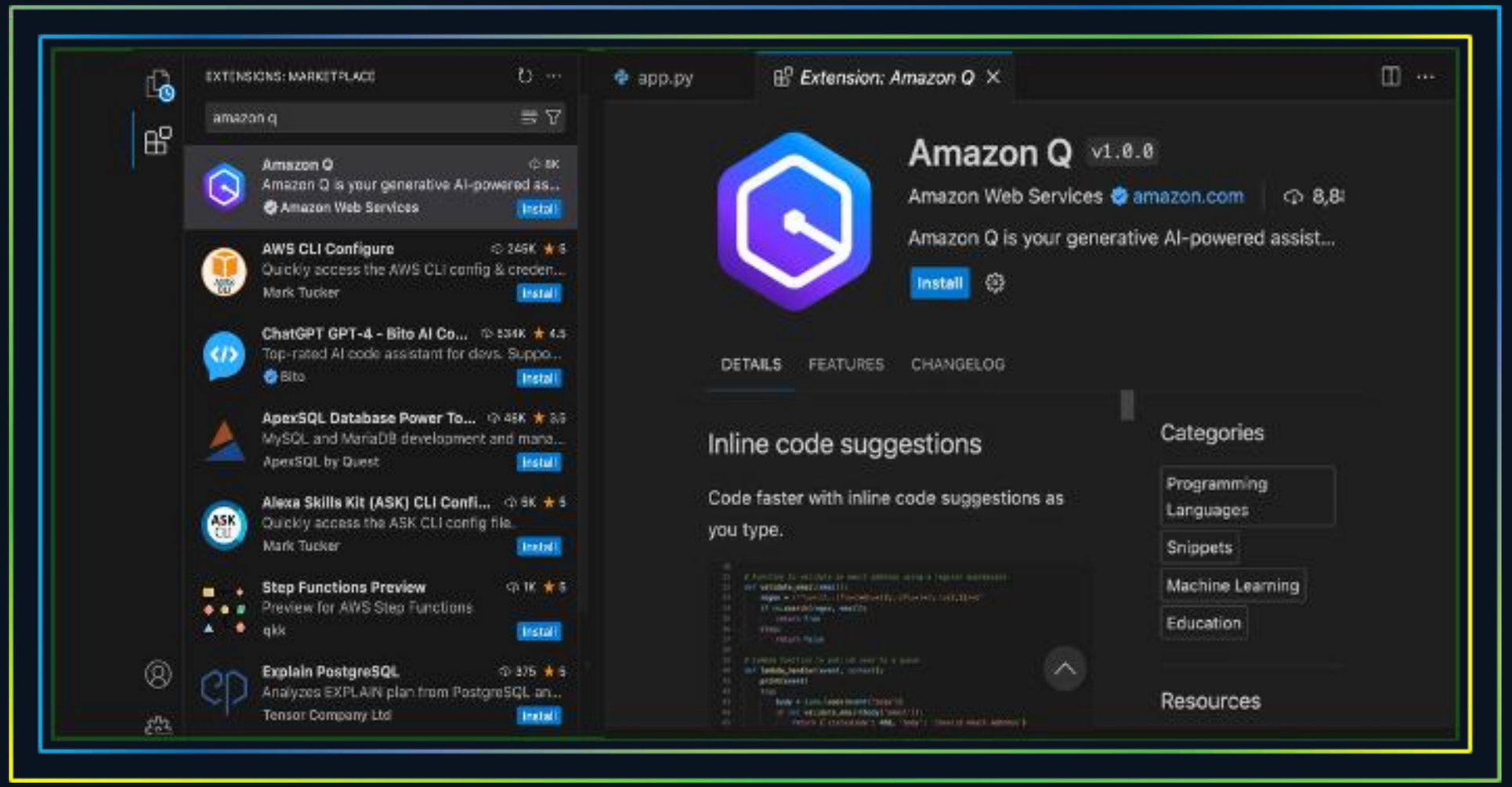
Cursor

Tabnine

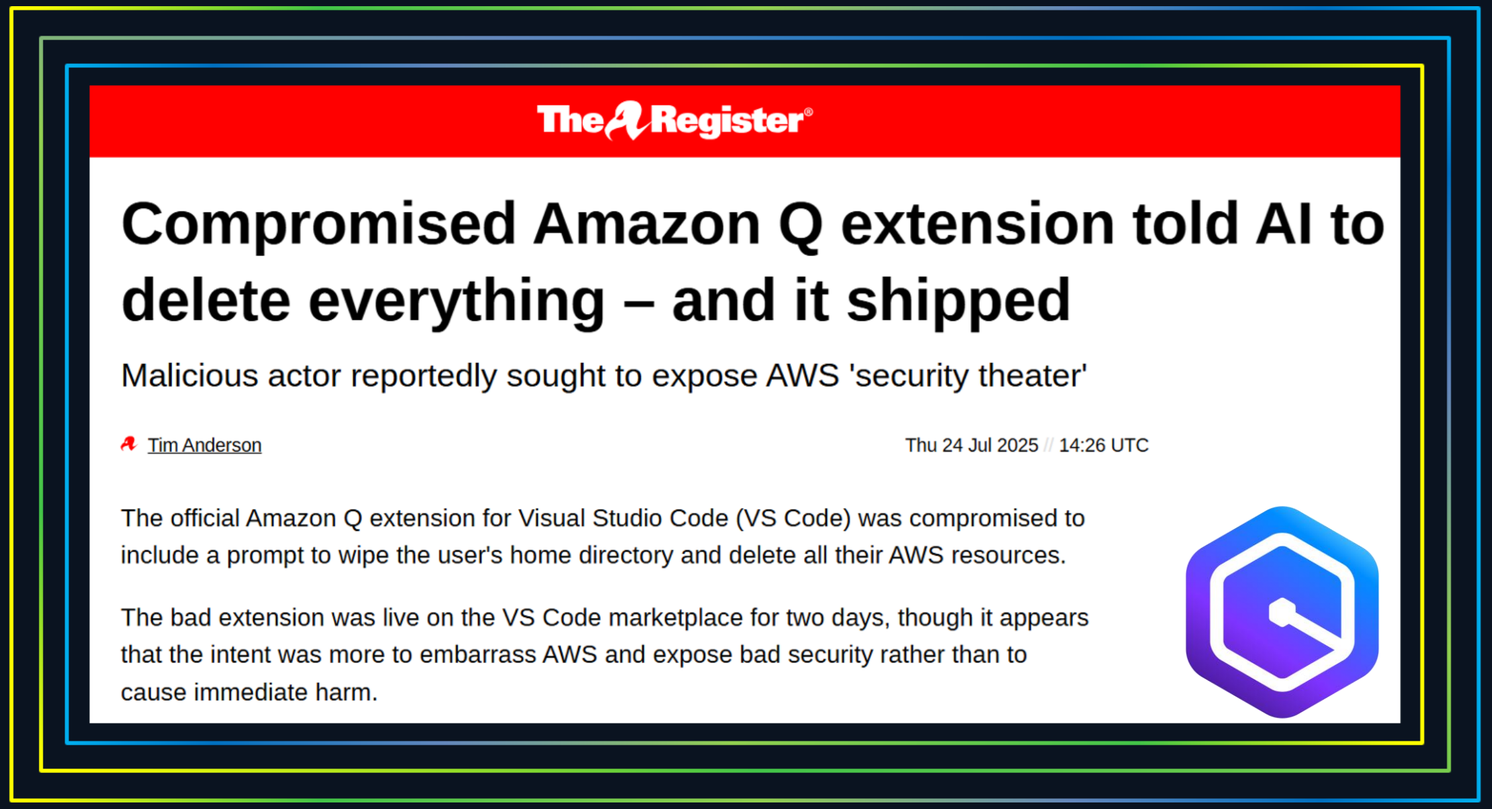
New is where

Threat Actors Thrive

Let’s Look at One Attack…



Let’s Look at One Attack…



The Register

Compromised Amazon Q extension told AI to delete everything – and it shipped

Malicious actor reportedly sought to expose AWS 'security theater'

Tim Anderson

Thu 24 Jul 2025 // 14:26 UTC

The official Amazon Q extension for Visual Studio Code (VS Code) was compromised to include a prompt to wipe the user's home directory and delete all their AWS resources.

The bad extension was live on the VS Code marketplace for two days, though it appears that the intent was more to embarrass AWS and expose bad security rather than to cause immediate harm.

How it Happened:

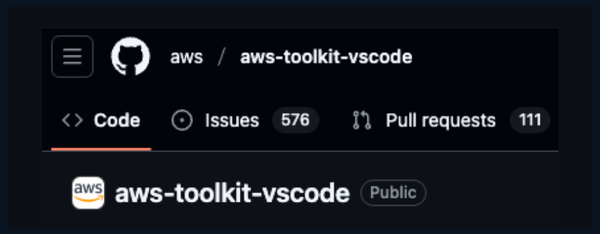
Attacker creates PR with malicious prompt to Amazon Q VS Code extension repository

PR accepted. New Version released

Developers obtain malicious Amazon Q VS Code extension

VS Code extension communicates with Amazon Q utilizing malicious prompt

Amazon Q returns malicious instructions to execute



SSC Threat Entry Points

NPM Package Hijack Overview ("Shai-Hulud" and co.)

26

Packages compromised

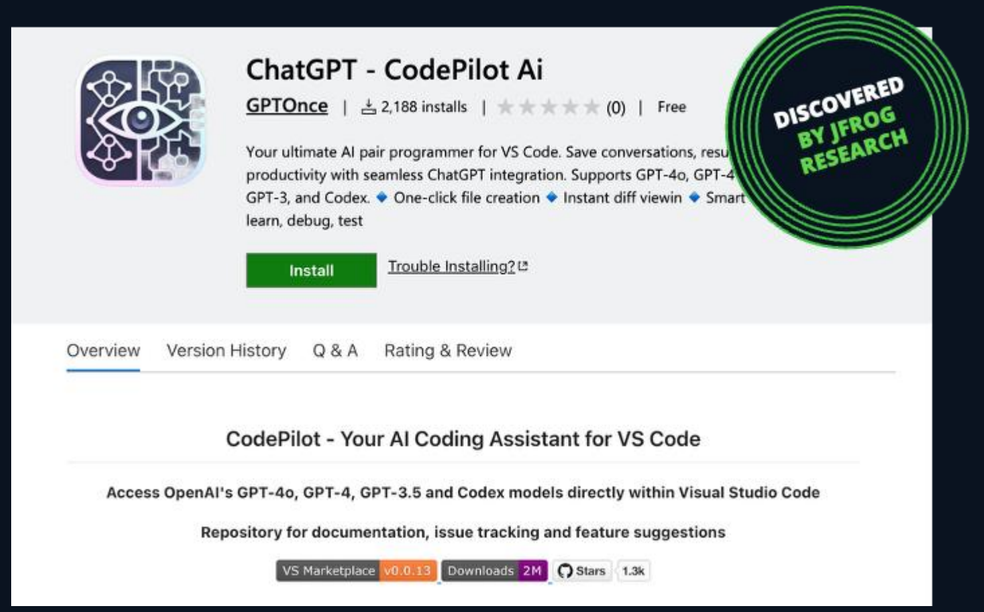
2M

Downloads of compromised package versions

- **JFrog was first to report 5 of the compromised packages**
- **Other packages identified by JFrog security scanners and marked as ‘malicious’ within just hours**
- **Several JFrog customers remained seamlessly protected with Curation blocking the risk**

Attackers injected malicious code to intercept and divert crypto currency transactions

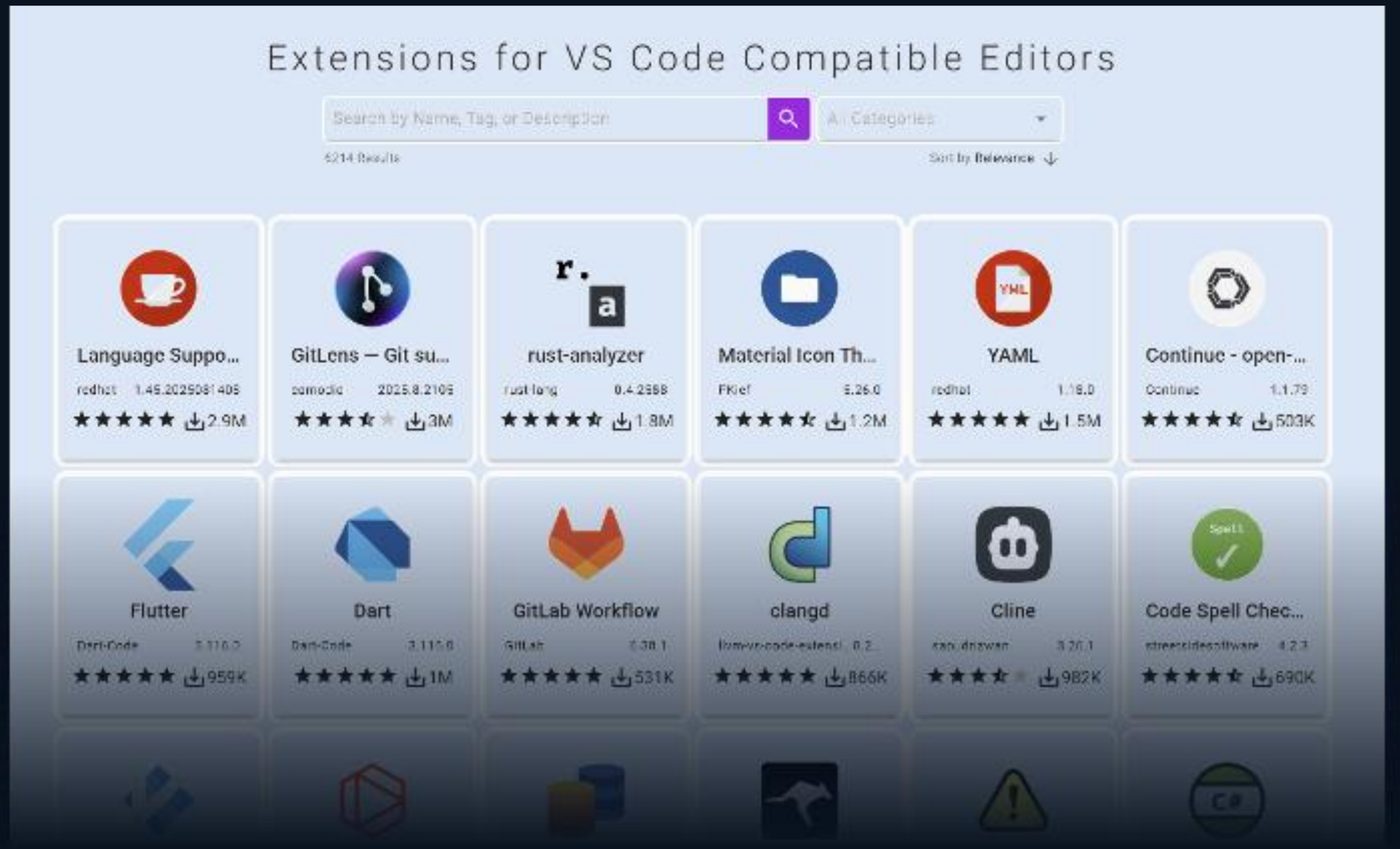
Yet another Malicious NPM Package....

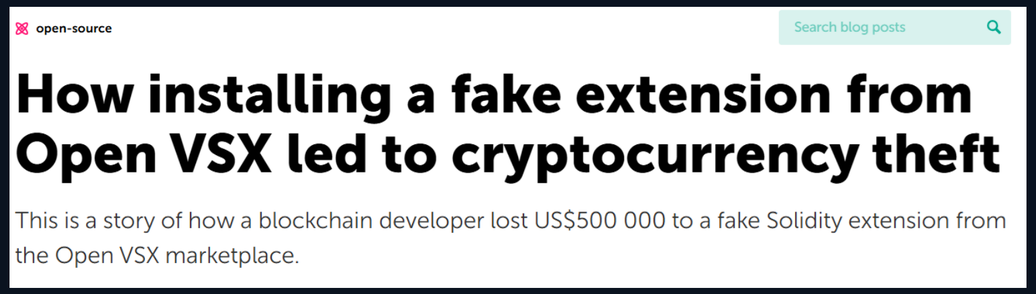


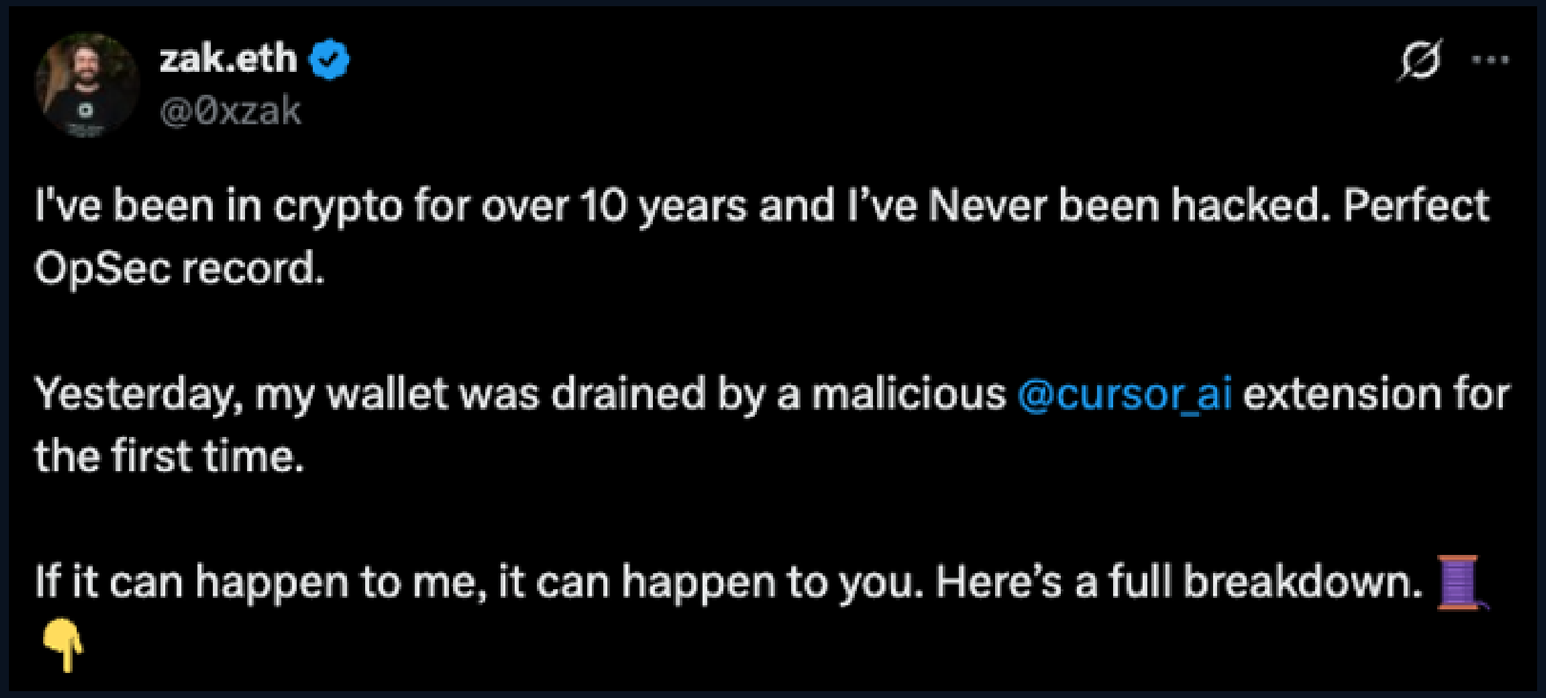
Same malware, same attacker - **Now on VSCode**

⚠ But this is not just a VSCode issue

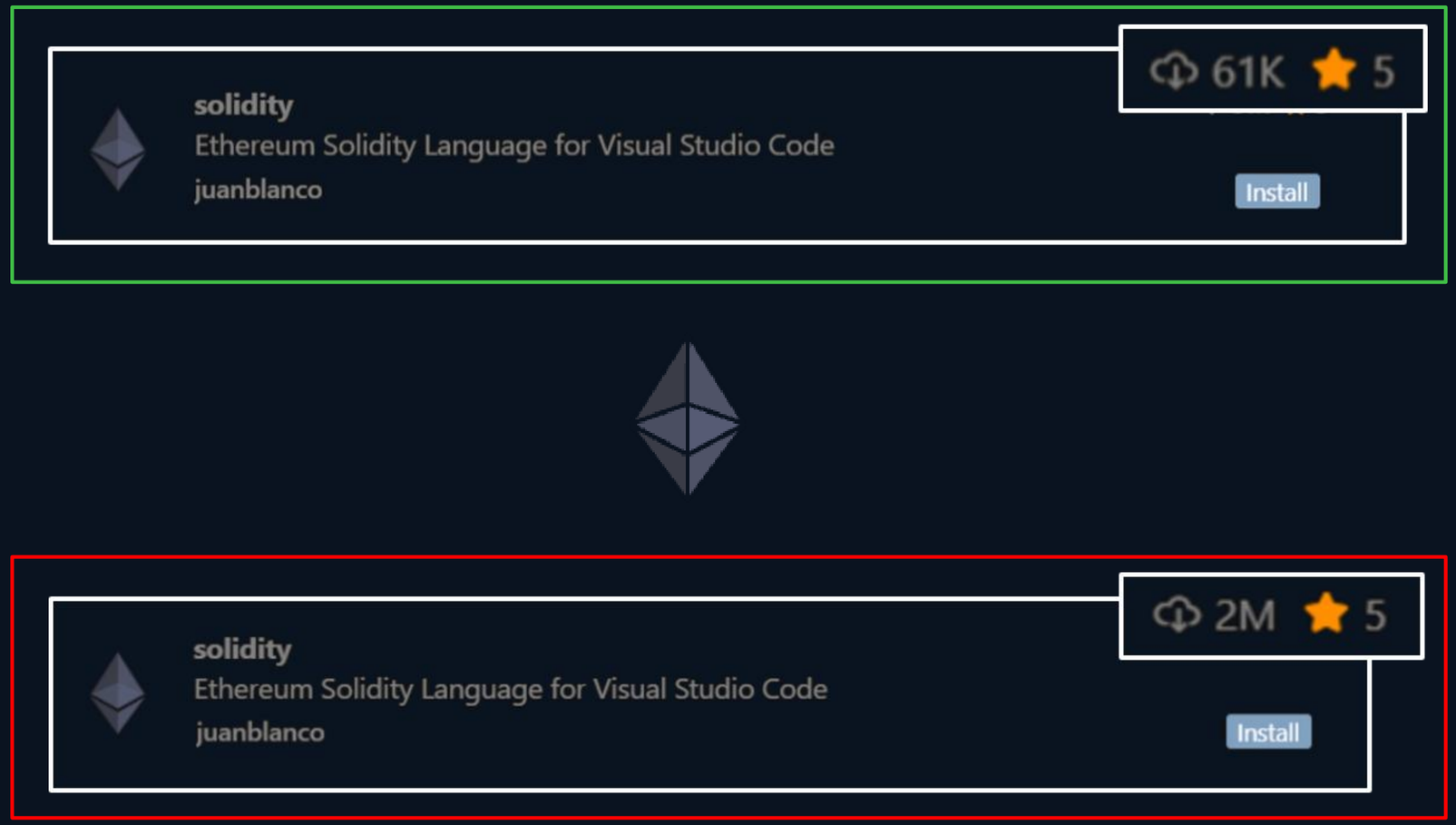
But this isn’t just for VSCode...




open-source
Search blog posts
How installing a fake extension from Open VSX led to cryptocurrency theft
This is a story of how a blockchain developer lost US$500 000 to a fake Solidity extension from the Open VSX marketplace.


zak.eth
@0xzak
I've been in crypto for over 10 years and I've Never been hacked. Perfect OpSec record.
Yesterday, my wallet was drained by a malicious @cursor_ai extension for the first time.
If it can happen to me, it can happen to you. Here's a full breakdown. 🧵
👇

Spot the Malicious Extension



Why do SCA tools SUCK?!

(at detecting malicious packages)

Why SCA Sucks (at detecting malicious packages)

JFrog Security ✓
@JFrogSecurity

Strong piece from SourceCodeRed sourcecodered.com/sca-sucks/

If traditional SCA struggles to spot malicious packages, what sets JFrog's malicious-package detection apart?
Our approach combines in-house scanners, curated public databases focused specifically on malicious packages (for example, OpenSSF), and continuous monitoring of attack activity.
That proactive, layered strategy has let us identify true "zero-day" malicious packages in the wild, including:
PyPI: mcp-runcmd-server

1. research.jfrog.com/post/3-malicio ...

PyPI: soopsocks

1. research.jfrog.com/post/check-you ...

npm: toolkdvv

1. jfrog.com/blog/malicious ...

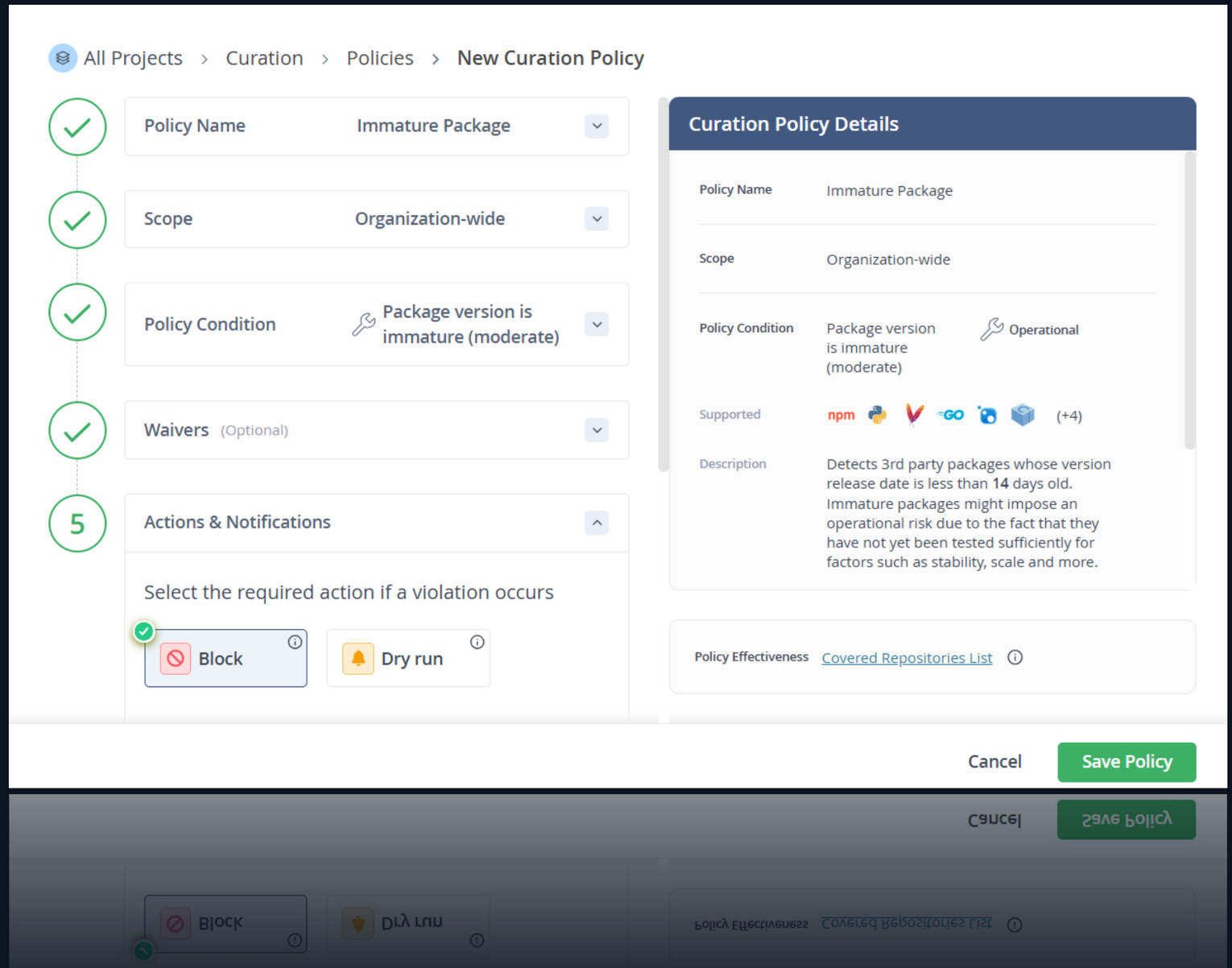
The Solution

We Need to
Curate the Packages used
in + the Tools used to
Develop Software

Prevent The Next Attack With JFrog Curation

JFrog's Security Research Team Investigated the Lifespan of Hijack Attack

- Point Developers to Artifactory
- Set policies to eliminate hijack attack risk
  Block new/immature packages (based on age)
- Make "Compliant Version Selection" effortless
  Divert package managers to mature versions for a seamless dev experience



Live Audit

12 Month period

~110k New Packages
Downloaded

4
Malicious

1,253
CVEs

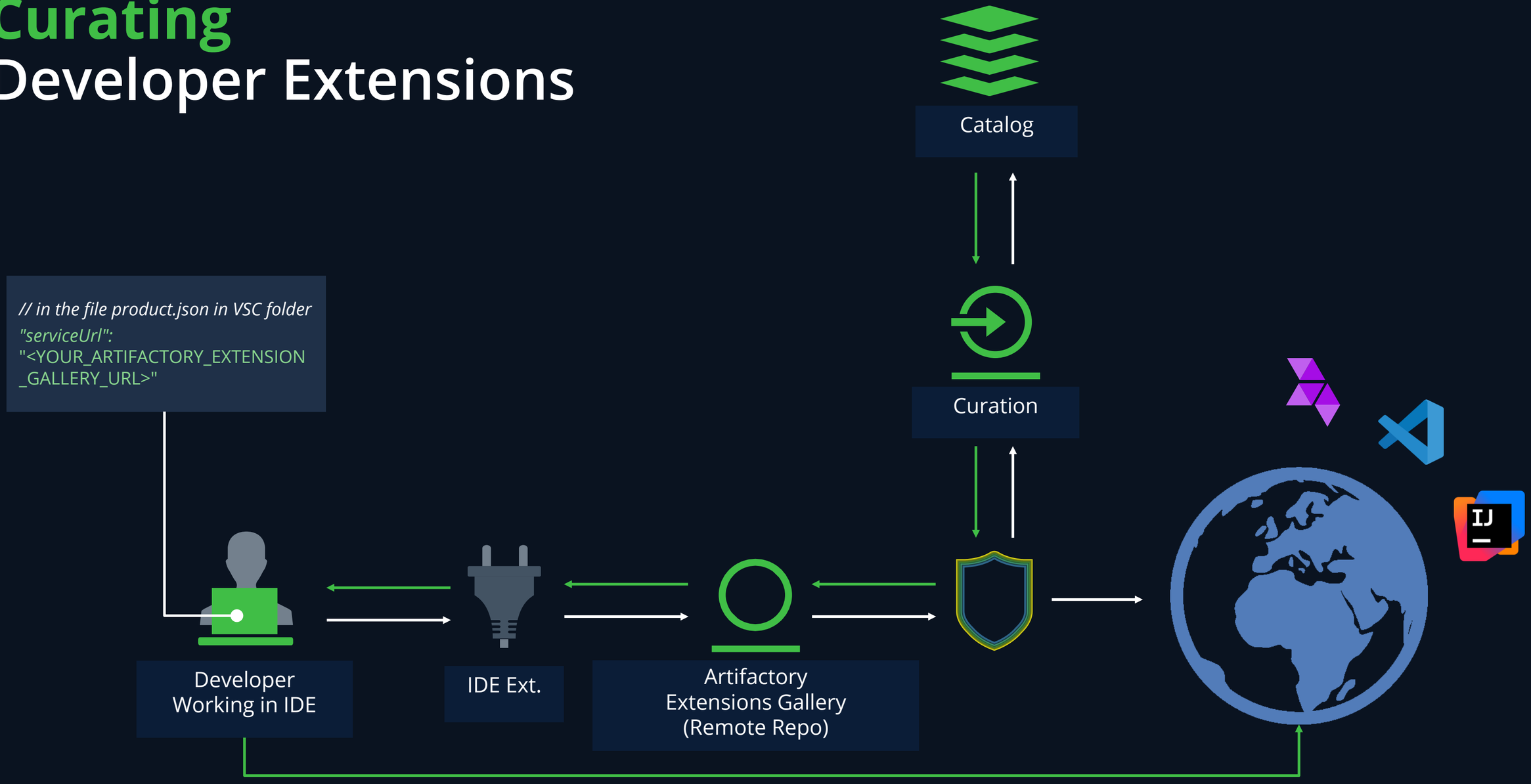
3,803
Licenses violations

1,021
Operational Risks

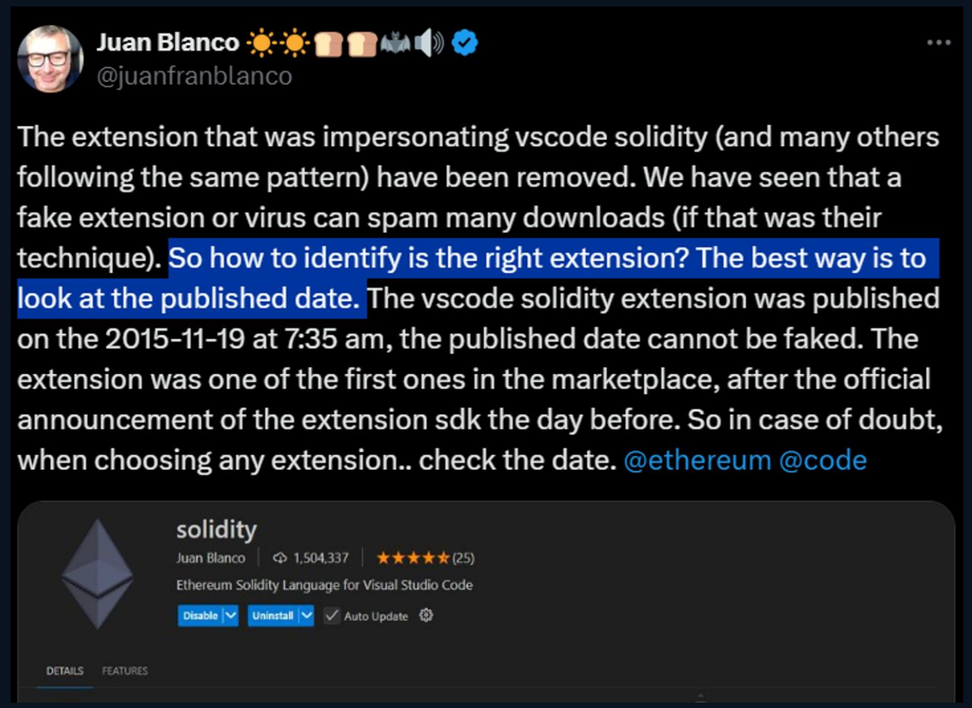
11,000
hours of remediation saved

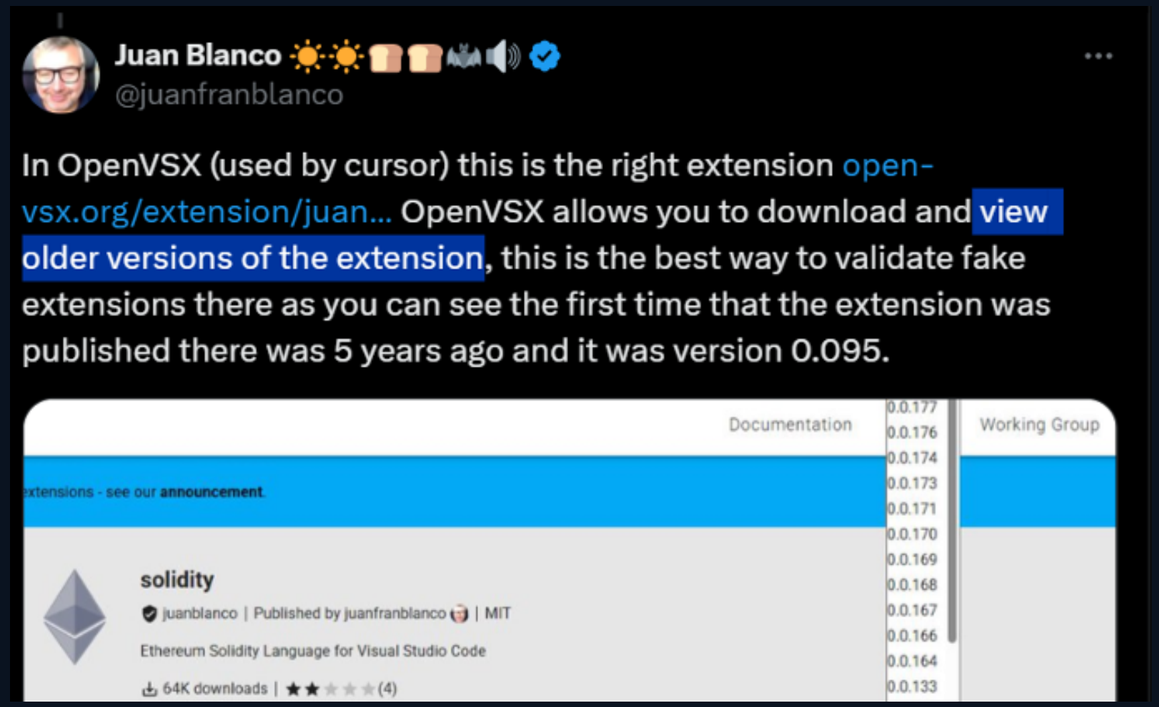
Curation Works.

Curating
Developer Extensions



The Challenge

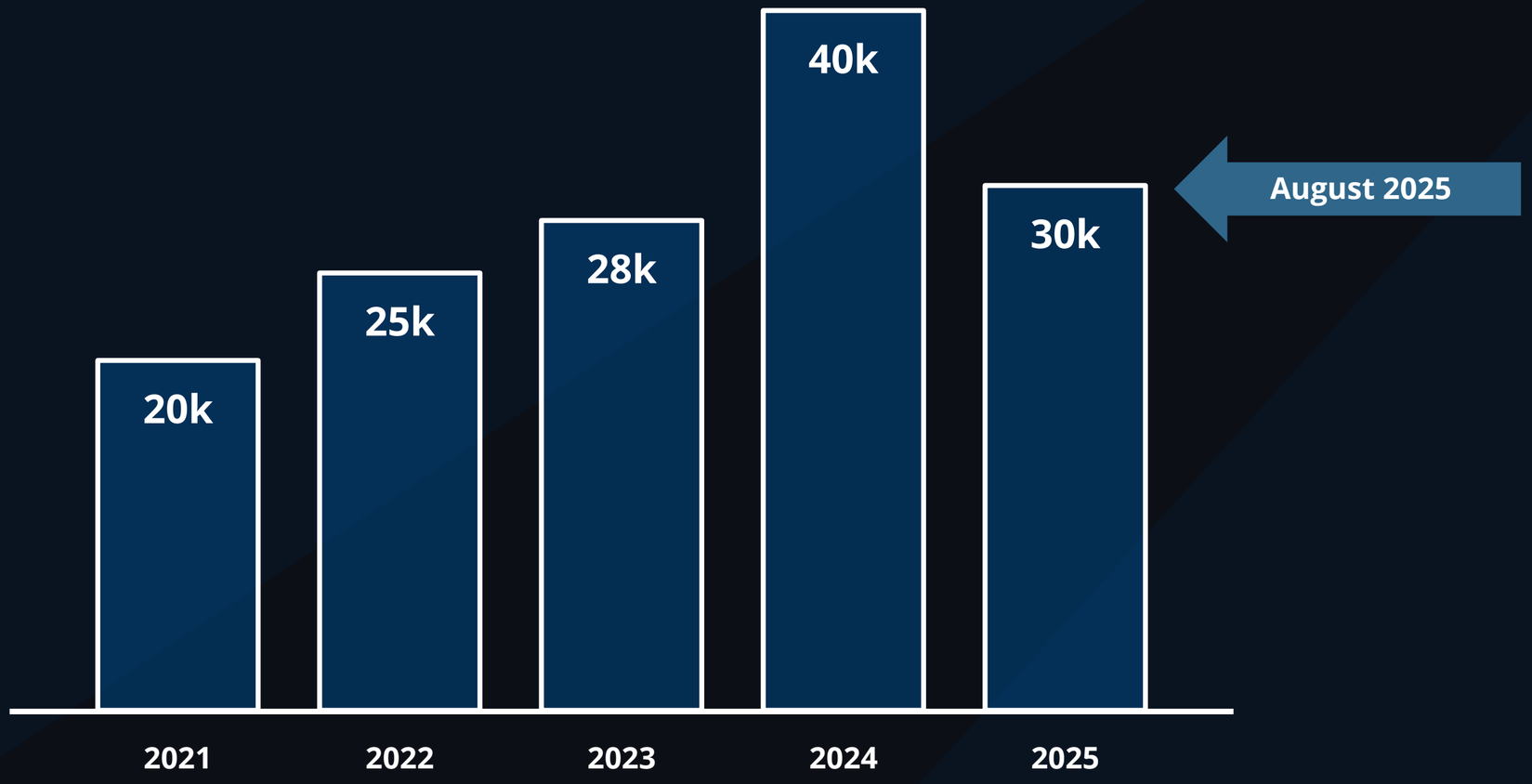





While Attackers are
Targeting the New

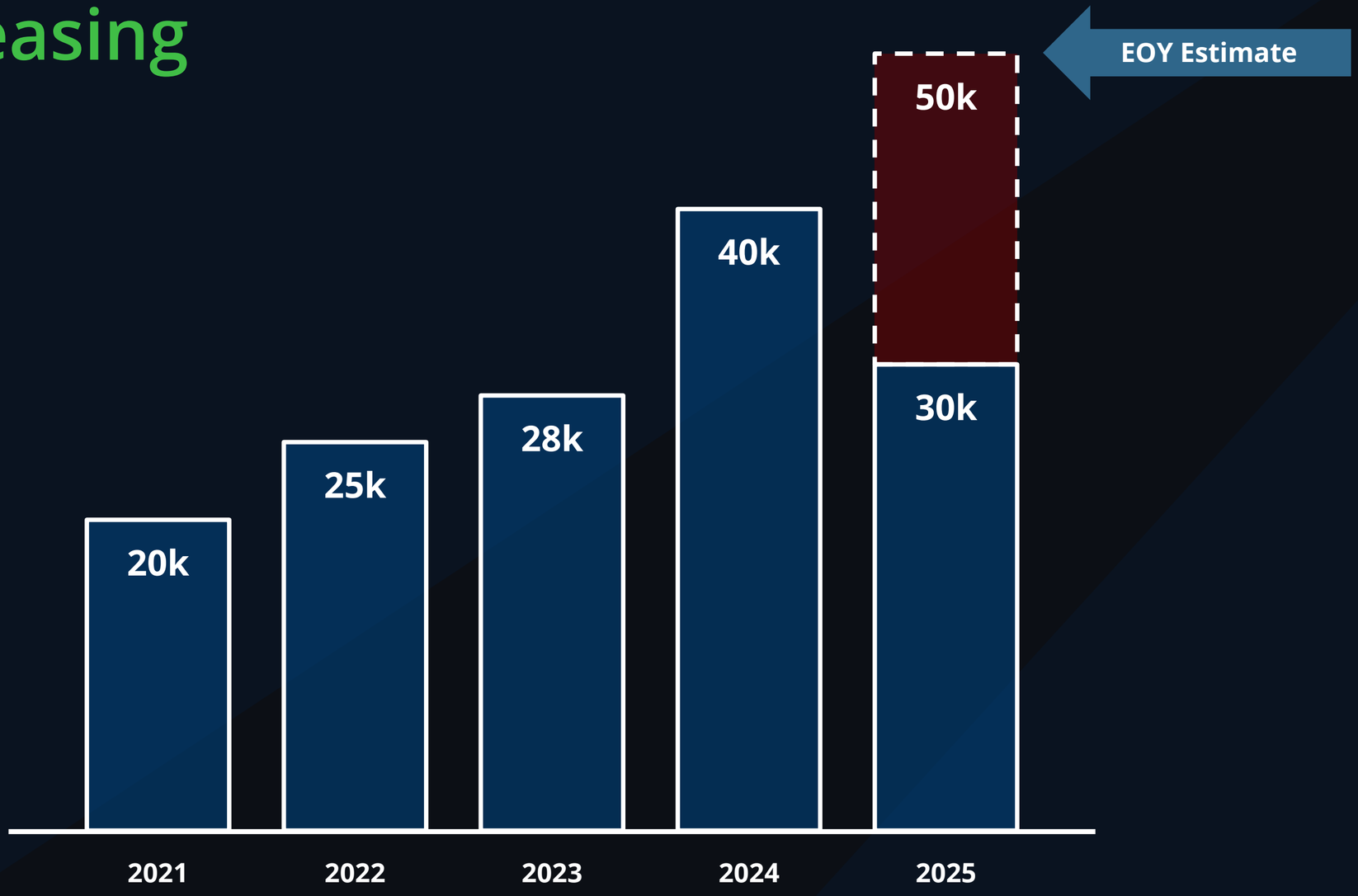

Developers are
Swamped by the Old

Number of CVEs
are Ever Increasing



NIST NATIONAL VULNERABILITY DATABASE NVD

Number of CVEs
are Ever Increasing



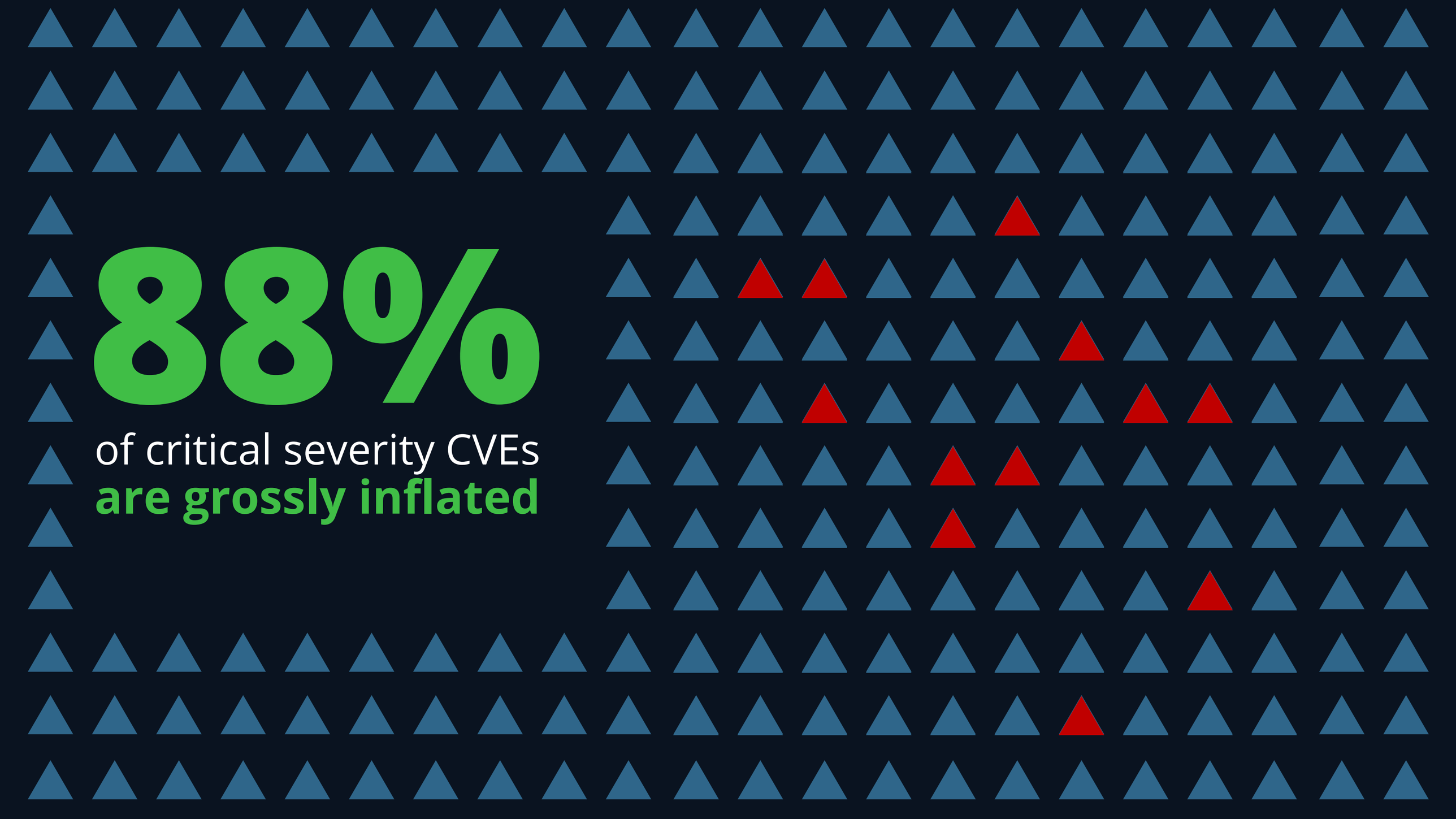
NIST NATIONAL VULNERABILITY DATABASE NVD

SLA times for Fixing CVEs are Constantly Decreasing

We need to focus on

CVEs

that are Critical
that Can be Exploited
and Run in Production


88%
of critical severity CVEs
are grossly inflated

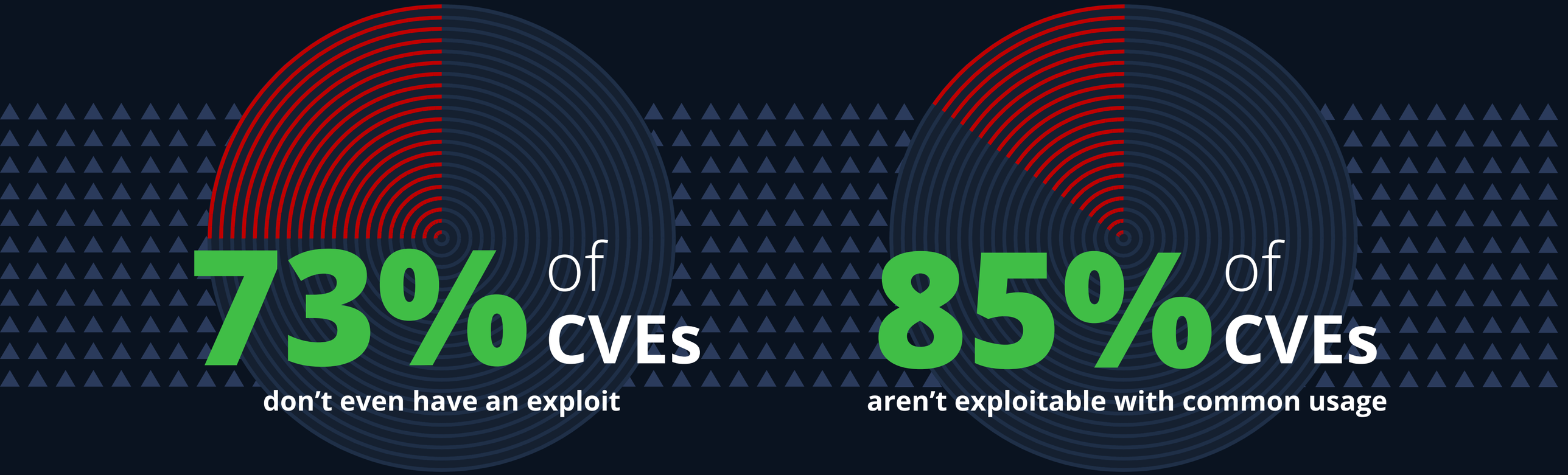
Exploitation Requires the Victim to
Download and Execute an Untrusted Malicious Package

1 Out of 3
JFrog Customers

We need to focus on

CVEs

that are Critical
that Can be Exploited
and Run in Production


73% of CVEs
don’t even have an exploit
85% of CVEs
aren’t exploitable with common usage

CVE Contextual Analysis

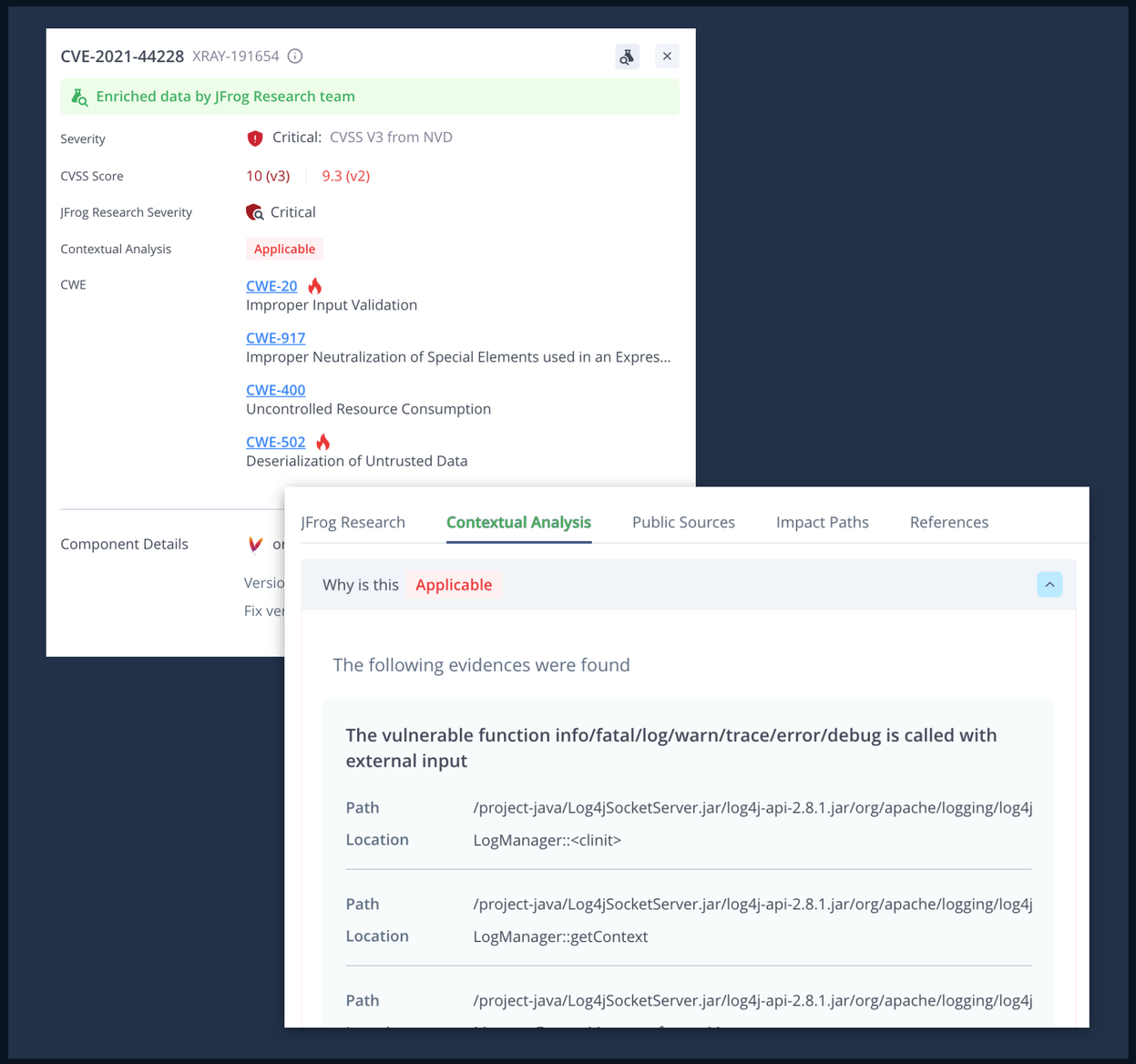
What Does it Do?

- Flags vulnerabilities that are **applicable** and **exploitable** in your organization
- Provides clear and **research-backed remediation guidance**

Why is it Important?

- Eliminates noise by focusing on **relevant vulnerabilities only**
- Improves **remediation efficiency**, eliminates wasted effort

88% Critical and 57% High CVE scores are not as severe as indicated by the CVSS score

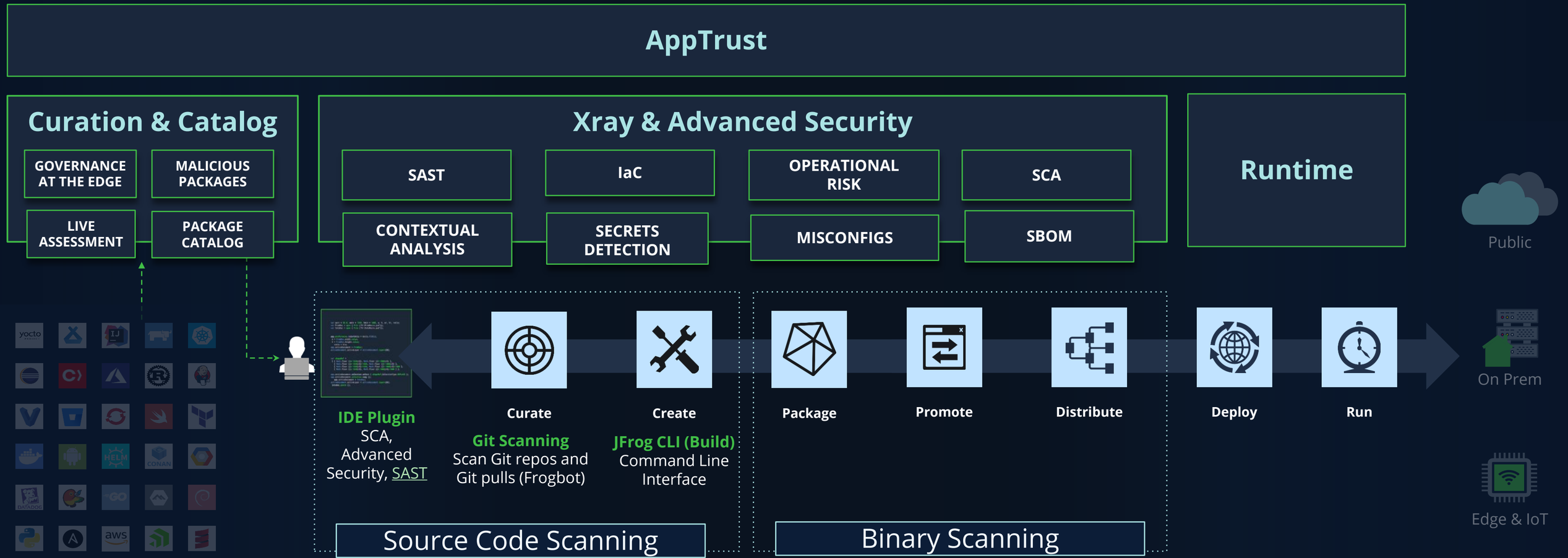


We need to focus on

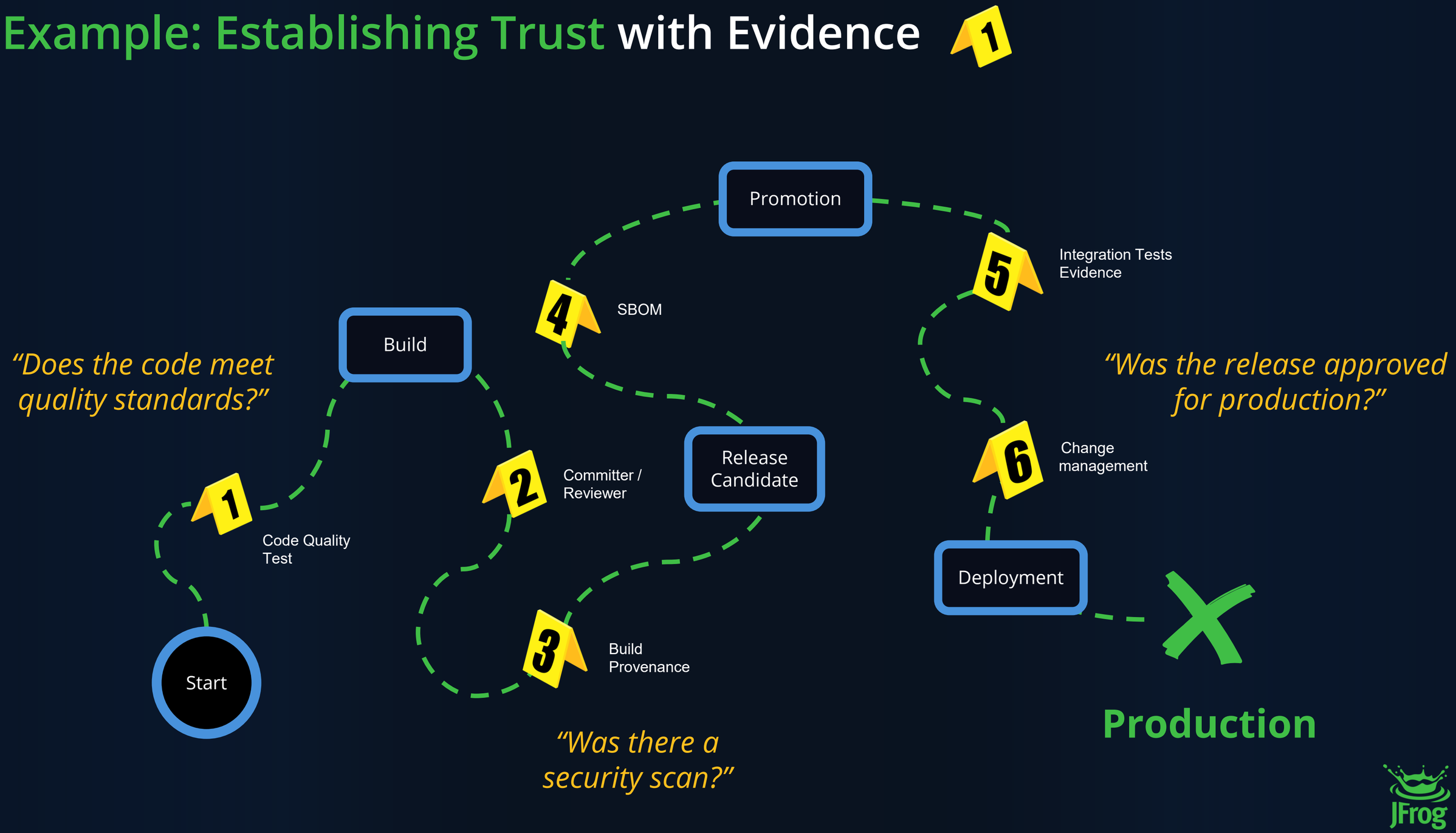
CVEs

that are Critical
that Can be Exploited
and Run in Production

End-to-end Security with JFrog



Example: Establishing Trust with Evidence



We Need to Focus On

CVEs that are Critical
that Can be Exploited
and Run in Production

Thank You