

npm install

SOFTWARE SUPPLY CHAIN

Who signed off on this **package?**

AI can suggest dependencies. People still have to own them.

Who signed off on this package?

AI can find, compare and install dependencies faster than any team. We think people should still own at least 90% of the decision about what goes into your software, and the evidence backs that up.

KEY TAKEAWAYS

1. Adding one npm package means trusting about 79 other packages and 39 maintainers nobody on your team picked.
2. In 2025, 84% of developers used or planned to use AI tools. Only 3% said they highly trust the output.
3. AI coding models still name packages that don't exist on npm or PyPI. Anyone can register those names for free.
4. The largest attacks of 2024 and 2025 hit packages people already trusted. Tools can flag risk. The approval has to belong to someone who understands the system.

Every install is a hiring decision

When you hire a dev partner, you also hire every library they choose. Most clients never see that list, and it is longer than they would guess. A USENIX Security study of npm found that installing an average npm package introduces an implicit trust on 79 third-party packages and 39 maintainers. Sonatype reported in 2024 that every enterprise application has at least 180 third-party components on average.

None of that is a reason to avoid open source. Modern software is built on it. It does mean that adding a package is an architectural decision, and someone has to be accountable for it.

79

packages you implicitly trust when you install one average npm package.

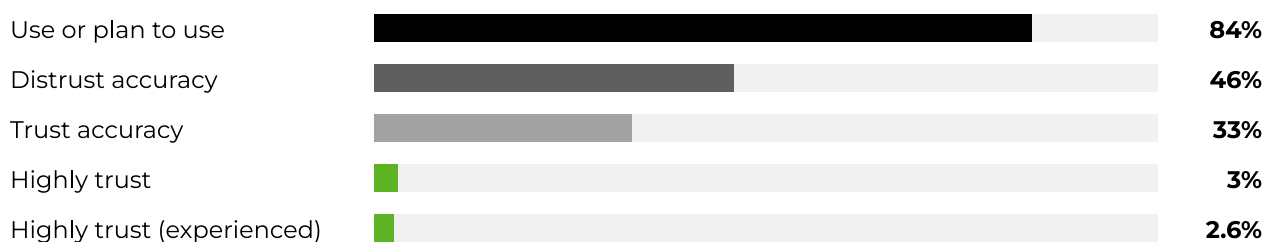
Zimmermann et al., USENIX Security 2019

Adoption went up. Trust went down.

Stack Overflow's 2025 Developer Survey drew more than 49,000 responses from 177 countries. It found that 84% of developers use or plan to use AI tools, up from 76% in 2024. Over the same year, the share who don't trust the accuracy of AI output rose from 31% to 46%.

Experienced developers were the most cautious, with the lowest "highly trust" rate (2.6%) and the highest "highly distrust" rate (20%). Our reading: the people closest to production already treat AI output as a draft to review, not an answer to ship.

Developers and AI tools, 2025



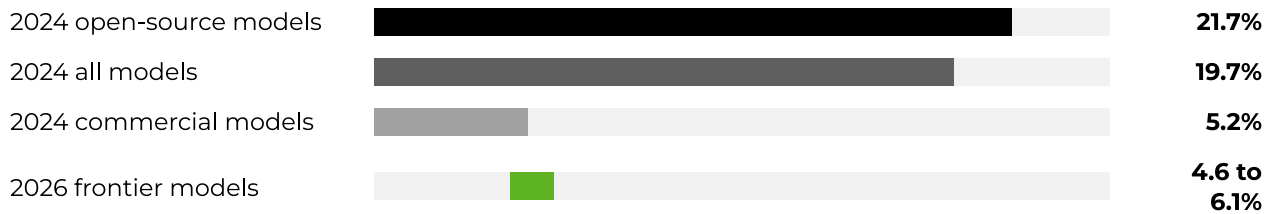
Source: Stack Overflow 2025 Developer Survey. Stack Overflow's own materials also cite 29% trust (down from 40%), a different cut of the same survey. We show the results-page figure.

When the suggestion doesn't exist

Researchers from UTSA, Virginia Tech and the University of Oklahoma generated 576,000 code samples from 16 models in Python and JavaScript. Of the packages those samples recommended, 19.7% were hallucinations, with 205,474 unique fake package names. Commercial models averaged at least 5.2%; open-source models averaged 21.7%. The paper appeared at USENIX Security 2025.

The attack this enables is called slopsquatting: register a name that models keep inventing, put malware in it, and wait. A 2026 replication on five newer frontier models found the problem narrowed to 4.62% to 6.10%, but not gone. It also found 127 names that all five models invented identically, 53 of which an attacker could still register. Mapping those hallucinations cost the researcher \$860.90 in API fees. That study is a single-author preprint, not yet peer reviewed, and it did not test agent setups that check the registry before installing.

Share of AI-recommended packages that don't exist



Sources: [Spracklen et al., USENIX Security 2025](#) (16 models, 576,000 samples); [Churilov, 2026 preprint](#) (5 models, 199,845 samples, range across models). Scale 0 to 25%.

An attacker doesn't need to fool your team. They only need your team to stop checking.

The worst attacks hit packages people already trusted

Fake names are one risk. Real names going bad is a bigger one. In March 2024, a backdoor rated CVSS 10.0 was [discovered in XZ Utils](#), a compression library used across Linux. The person behind it had [spent about two years building trust](#) as a contributor before becoming co-maintainer. A Microsoft engineer caught it because SSH logins on a test machine were [taking about 500ms longer than expected](#). A person noticed.

On September 8, 2025, one maintainer fell for a phishing email. Malicious versions of chalk, debug and 16 other packages followed, [representing 2.6 billion weekly downloads](#). A week later came Shai-Hulud, a self-replicating worm that [compromised over 500 packages](#), according to CISA. Its second wave in November [backdoored 796 unique npm packages](#) with over 20 million weekly downloads.

Underneath all of this, the volume keeps growing. Sonatype logged [704,102 malicious open source packages since 2019](#) as of August 2024, with 156% growth year over year. A 2026 roundup citing Sonatype's newest report puts the cumulative total at [over 1.233 million](#). We have not seen that figure in Sonatype's primary report, so read it as reported.

2 years

XZ Utils attacker building trust before the backdoor

2.6B

weekly downloads behind the chalk and debug compromise

796

npm packages backdoored by Shai-Hulud 2.0

Why 90%

The 90% is a stance, not a measurement. AI is good at the search: listing options, reading changelogs, running scanners, drafting the comparison. It should do that work. What it can't do is know your system: which services hold secrets, what the build runs at install time, how much a dependency's failure would cost you, and whether a twelve-line utility is worth importing at all.

As teams hand more of the typing to AI, the skill that's left is discernment. Choosing what not to add, knowing why something is there, and drawing the boundaries that limit damage when a trusted package turns. That is architecture, and it's how a team shows a client it understands what it built.

If you are hiring a dev partner, ask them who approves new dependencies and how. A good answer names a person and a process. "The tool picked it" is not an answer.

CHECKLIST

Before a package gets approved

- 01 Confirm it exists and is the real one.** Check the exact name on npm or PyPI, its repository, publisher and age. Never install a name straight from an AI suggestion.
- 02 Justify why it's needed.** Write one sentence on what it does that we can't do in a few lines ourselves.
- 03 Look at what comes with it.** Review the transitive dependency count, maintainers, and any install scripts.
- 04 Pin it and lock it.** Commit the lockfile, pin versions, and treat every upgrade as a new approval.
- 05 Put a name on it.** One person approves the addition in review and can explain it to the client.

SOURCES

1. Zimmermann, Staicu, Tenny, Pradel. "Small World with High Risks: A Study of Security Threats in the npm Ecosystem." USENIX Security 2019. [usenix.org](https://www.usenix.org)
2. Stack Overflow. "2025 Developer Survey" press release, July 29, 2025. stackoverflow.co
3. Stack Overflow. 2025 Developer Survey, AI section. survey.stackoverflow.co/2025/ai
4. Spracklen et al. "We Have a Package for You! A Comprehensive Analysis of Package Hallucinations by Code Generating LLMs." USENIX Security 2025. [arXiv:2406.10279](https://arxiv.org/abs/2406.10279) · [artifact](#)
5. Churilov. "The Range Shrinks, the Threat Remains: Re-evaluating LLM Package Hallucinations on the 2026 Frontier-Model Cohort." Preprint, 2026. [arXiv:2605.17062](https://arxiv.org/abs/2605.17062)
6. "XZ Utils backdoor" (CVE-2024-3094). [Wikipedia](https://en.wikipedia.org) · Safeguard, "Lessons from the XZ Utils Backdoor." safeguard.sh
7. Jscrambler. "The npm Chalk and Debug Attack." September 2025. jscrambler.com · Semgrep security alert. semgrep.dev
8. CISA. "Widespread Supply Chain Compromise Impacting npm Ecosystem." September 23, 2025. cisa.gov
9. Datadog Security Labs. "The Shai-Hulud 2.0 npm worm." November 2025. datadoghq.com
10. Sonatype. "10th Annual State of the Software Supply Chain." October 2024. sonatype.com · via CSO Online. csoonline.com
11. AppSec Santa. "Supply Chain Attack Statistics 2026" (secondary, citing Sonatype 2026). appsecsanta.com