

Security Spotlight Theater

WELCOME TO



DASH
BY DATADOG

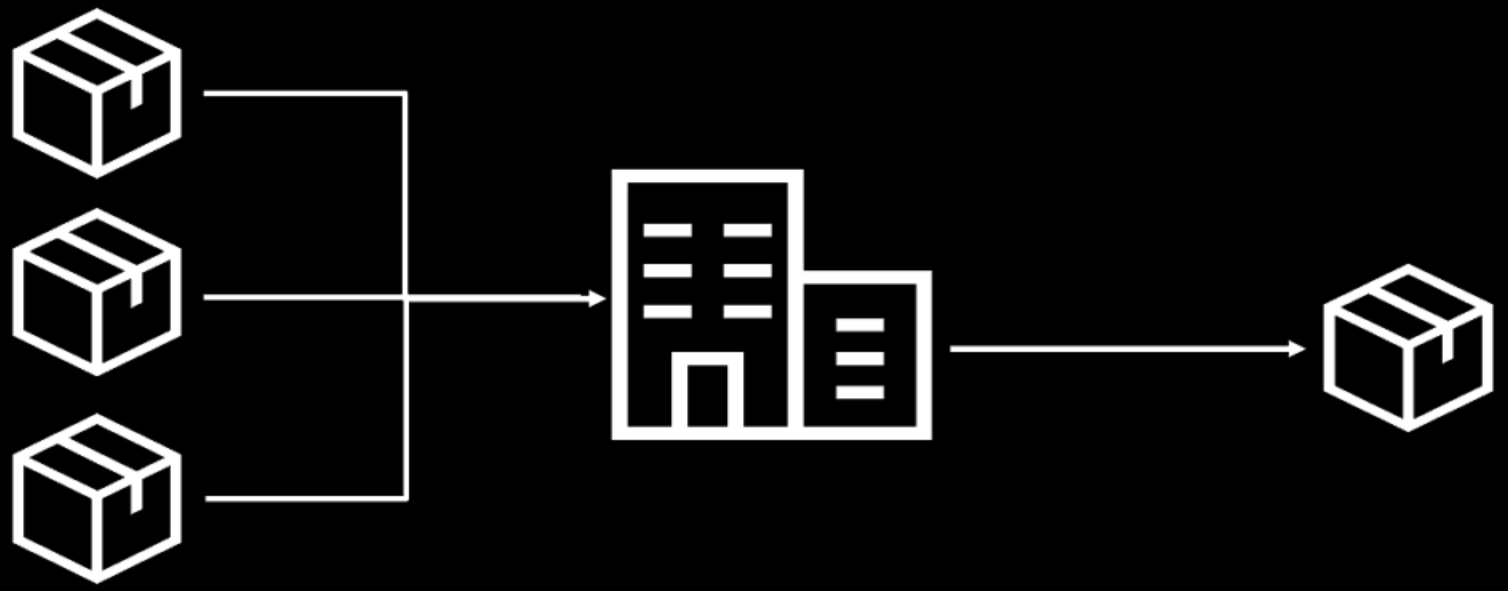
STARTING SOON



Inside the Supply-Chain Attacks of 2026

Key Learnings and Quick Wins

The software factory



DASH

The software factory



DASH

Have you met your (software) supply chain?



app dependencies
npm, PyPI



Docker images



VM images



AI skills

DASH

npm in numbers



4M+

public packages



600B+

monthly downloads

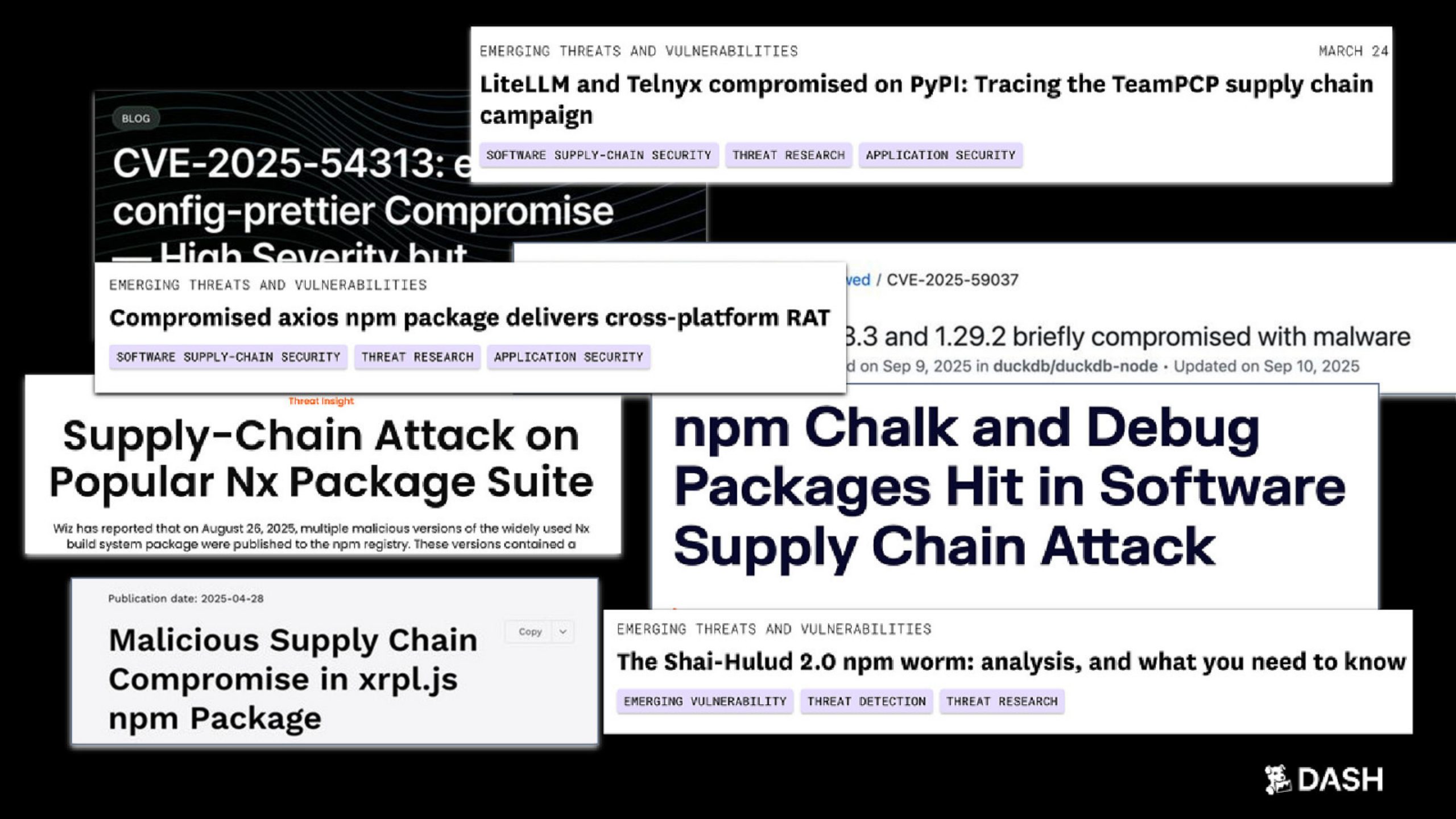


150k+

malicious packages

2026

the year of ~~Linux on the desktop~~
large-scale supply-chain attacks


EMERGING THREATS AND VULNERABILITIES
MARCH 24
LiteLLM and Telnyx compromised on PyPI: Tracing the TeamPCP supply chain campaign
SOFTWARE SUPPLY-CHAIN SECURITY
THREAT RESEARCH
APPLICATION SECURITY
BLOG
CVE-2025-54313: e
config-prettier Compromise
— High Severity but
EMERGING THREATS AND VULNERABILITIES
Compromised axios npm package delivers cross-platform RAT
SOFTWARE SUPPLY-CHAIN SECURITY
THREAT RESEARCH
APPLICATION SECURITY
ved / CVE-2025-59037
3.3 and 1.29.2 briefly compromised with malware
d on Sep 9, 2025 in duckdb/duckdb-node · Updated on Sep 10, 2025
Threat Insight
Supply-Chain Attack on Popular Nx Package Suite
Wiz has reported that on August 26, 2025, multiple malicious versions of the widely used Nx build system package were published to the npm registry. These versions contained a
npm Chalk and Debug Packages Hit in Software Supply Chain Attack
Publication date: 2025-04-28
Malicious Supply Chain Compromise in xrpl.js npm Package
Copy
EMERGING THREATS AND VULNERABILITIES
The Shai-Hulud 2.0 npm worm: analysis, and what you need to know
EMERGING VULNERABILITY
THREAT DETECTION
THREAT RESEARCH
DASH



Daniel Maher

Senior Security Advocate & Researcher

?

year of the first documented
malicious npm package

2017

year of the first documented
malicious npm package

Malicious software packages in 2026

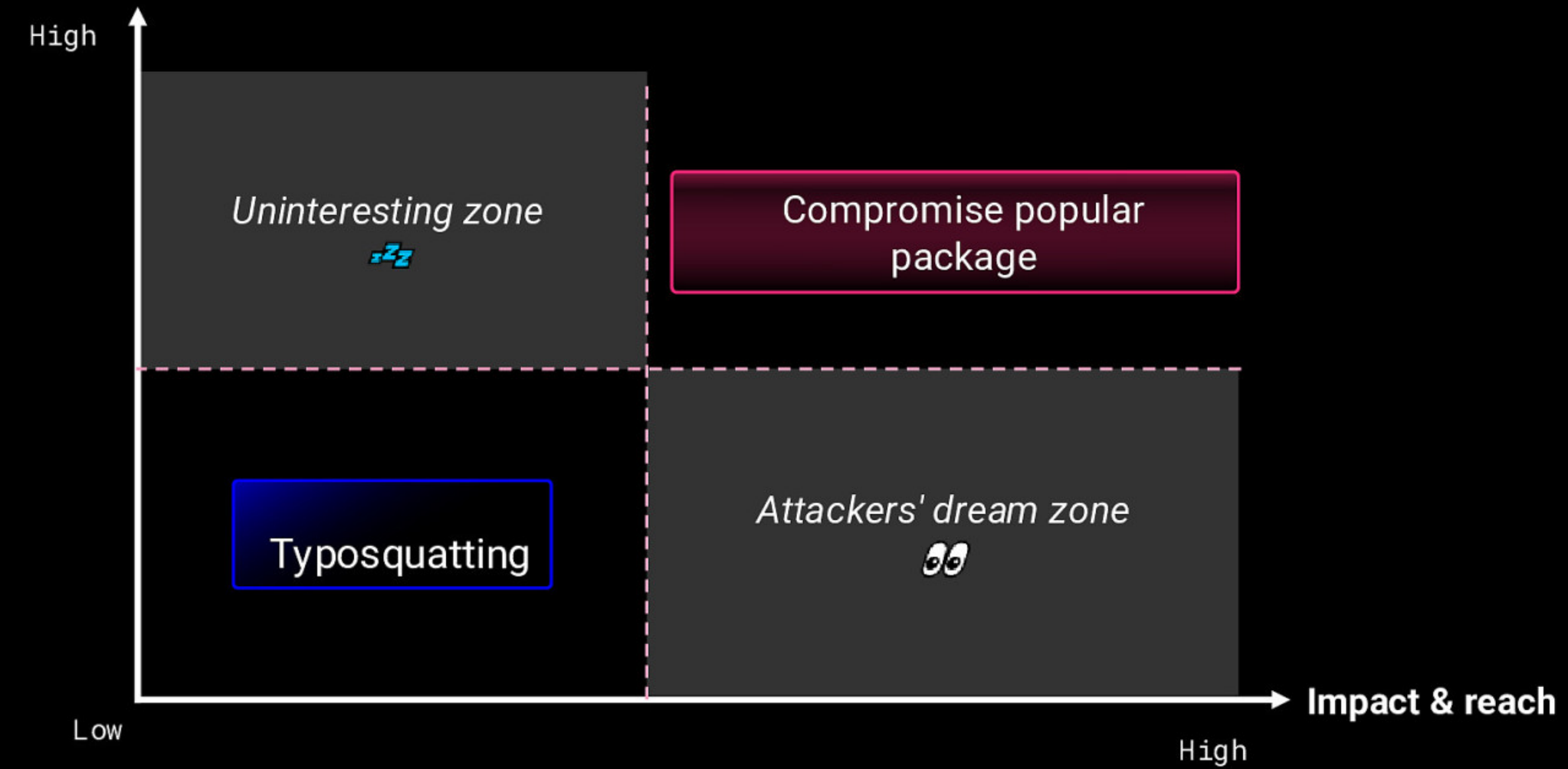
Malicious by design

Typosquatting
Brand impersonation
Popular keywords ("*ChatGPT*")

Compromised legitimate package

Compromised maintainer account
Release CI pipeline abused
CI credentials leaked

DASH


Attack complexity
High
Uninteresting zone
Compromise popular package
Typosquatting
Attackers' dream zone
Low
High
Impact & reach

DASH

Compromised legitimate packages



Large user base

`react: 100M+ weekly`
`axios: 100M+ weekly`
`lodash: 150M+ weekly`



Spread fast

50% of organizations use a dependency < 24h old[1]

1: State of DevSecOps, Datadog, 2026

Maintainer compromise

how attackers steal npm accounts

Adversary-in-the-Middle (AiTM) phishing

How it works

Attacker proxies auth through a lookalike npm domain
Captures valid session cookie — bypasses SMS/OTP 2FA
One compromised maintainer = thousands of victims

The lookalike infrastructure

npmjs.help npmjs.wtf npmjs.pro npmjs.us
npm email addresses are public — easy to target
Session cookie gives full npm publishing rights

xrpl (semantic versioning considered harmful)

Four staged versions

v4.2.1 — crude fetch(), only build files touched
v4.2.2/3 — expanded scope, refactored code
v4.2.4 — polished, malicious payload fully hidden

Why semantic versioning amplifies this

^4.2.0 silently auto-accepts new patches
50% of orgs install a dep released the same day
(Datadog 2026 State of DevSecOps)

The maintainer compromise two-step

Step 1: Gain access

AiTM phishing captures a valid session cookie
Bypasses all standard 2FA — undetectable in real time
Attacker controls the npm publishing account

Step 2: Maximize blast radius

Inject malicious code into a new patch version
Semantic versioning auto-delivers it to consumers
Steal secrets, deploy RAT, or establish persistence

?

year of the first documented
npm worm

2025

the npm worm era

Shai-Hulud 1.0 (September 2025)

How it self-propagates

Finds npm tokens in .npmrc and environment variables
Lists victim's packages via registry API
Publishes backdoored patch versions, then spreads

Secrets + impact

TruffleHog + env vars + cloud credentials stolen
500+ packages compromised
Some victims had 2M+ weekly installs

Shai-Hulud 2.0 (November 2025)

Significantly evolved

Cloud Secrets Manager (AWS/GCP/Azure) targeted
Self-hosted GitHub runner planted for persistence
GitHub Actions exfiltrates on every Discussion event

No token, no problem

Searches GitHub Archive for prior victims' tokens
700+ packages compromised — no central IOC list
Shai-Hulud 3.0 (Dec): minor variant, same codebase

axios supply chain (March 2026)

How the attack worked

Maintainer PC compromised via social engineering
plain-crypto-js@4.2.1 injected into package.json
Post-install hook deployed a cross-platform RAT

Impact

axios: 70M+ weekly downloads
Live ~3 hours; thousands of endpoints hit the command & control platform
Attributed to *Sapphire Sleet* (North Korea)

Sandworm (February 2026)

Persistence: two layers

Installs MCP server — injected into Claude Code, Cursor
Prompt injection in tool descriptions drives execution
Git hooks in all repos + init.templateDir — permanent

Advanced evasion

DGA for C2 + DNS tunneling as fallback
Calls local Ollama to rewrite own code per victim
Each instance unique — evades signature detection

The attacker’s toolkit

The four techniques

Version poisoning — code injected into a legitimate patch
Token and credential theft — npm tokens, cloud secrets
Post-install hooks — automatic execution at install time

Heuristic obfuscation

Sandworm rewrote its own code per victim
Variable renaming, control flow rewrites, decoy insertion
No two instances identical — signatures fail by design

Quick wins

what you can do this week

If you publish packages

Phishing-resistant 2FA

Switch to FIDO2 / WebAuthn: YubiKey or passkeys
Key is origin-bound — AiTM cannot replay it
Technically defeats the maintainer-phishing vector

npm Trusted Publishing

OIDC short-lived tokens via GitHub Actions
No long-lived token stored anywhere
Two lines of YAML — works with GitHub and GitLab

If you consume packages

Slow down updates

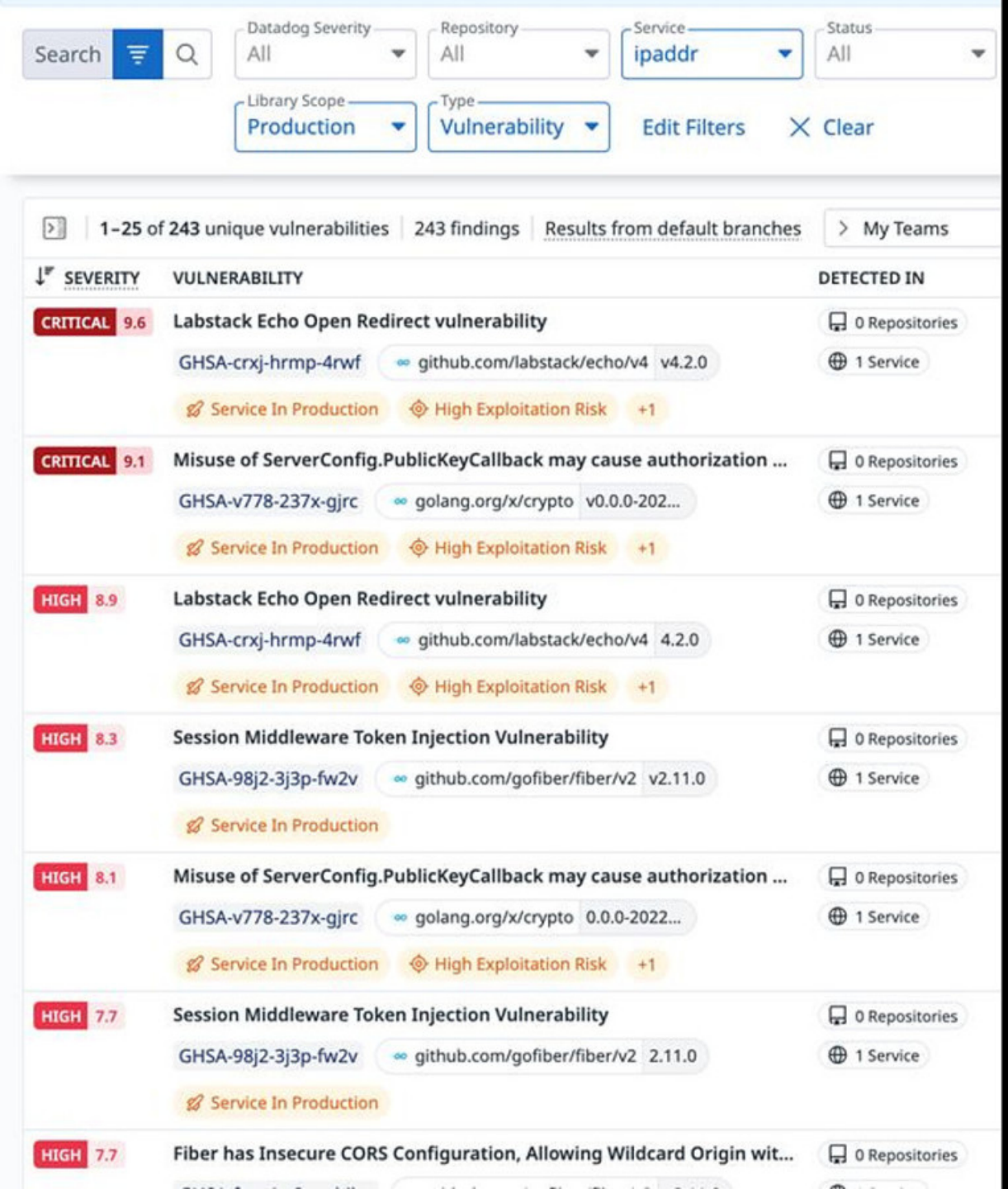
minimumReleaseAge / cooldown — delay N days
Closes the zero-hour infection window
SCFW wraps npm/pip, checks osv.dev at install time

Reduce the execution surface

npm config set ignore-scripts true
Disables postinstall hooks — how most worms fire
Default in pnpm v10+ (consider switching!)

How Datadog helps

static, runtime, and research



Datadog Code Security

Static analysis detects malicious packages at PR time

Runtime protection flags suspicious behaviour

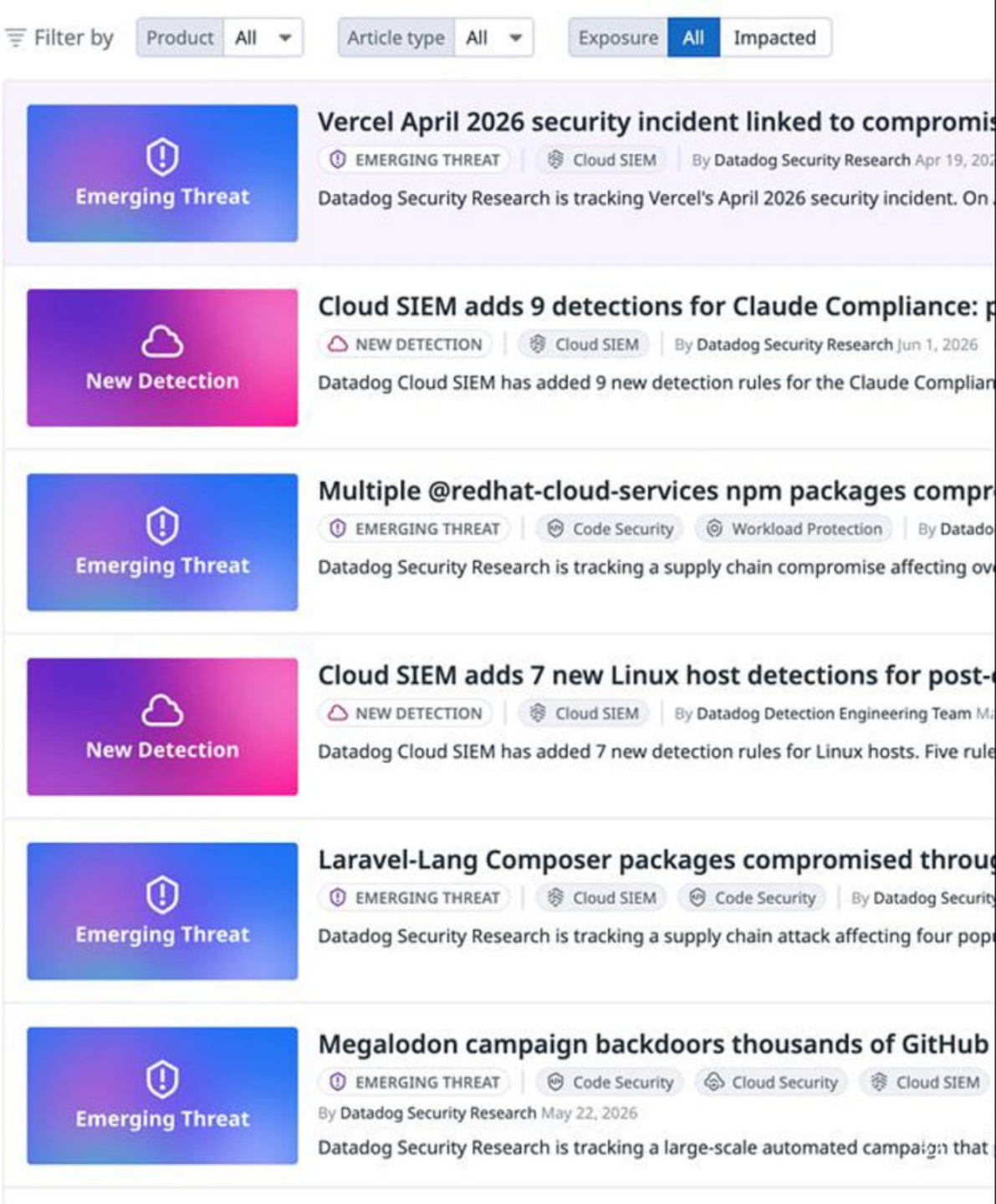
Contextualised alerts — severity + reachability

Datadog Security Research

Datadog Security Labs publishes malicious package research

MCP / Cmd+I brings alerts into your IDE in context

Tells you about malicious packages while you code



What's next

the attack surface is shifting

AI is the new attack surface

Attacks in the AI era

Postinstall hook fires, invokes local coding agent
Agent executes attacker commands — looks like dev activity
Sandworm already demonstrated MCP-layer persistence

MCP servers and AI skills

Same trust model as npm — install and run
Poisoned MCP server = attacker inside your dev environment
AI-heavy orgs are high-value targets (LiteLLM: 36% of envs)

La plus ça change…

Social engineering still wins

100% of 2025-26 maintainer compromises started with phishing
Public emails + high trust = easy targets
The attack is on people, not code

Classic defenses still apply

Inventory everything: SBOM, deps, MCP servers installed
Least-privilege: scoped tokens, no long-lived credentials
Delay auto-updates. Verify before you trust.

Ship safer

github.com/DataDog/guarddog
docs.datadoghq.com/security/code_security
securitylabs.datadoghq.com