SafeDep

# What If npm install Could Say No?

Realtime Defense Against Malicious Packages

# How does a dev machine get owned?

*What we usually defend against*

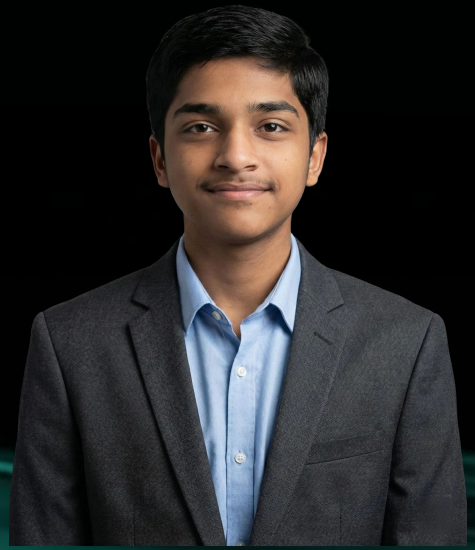
Phishing · USB · Extensions · Drive-bys

---

***But we never put this on the list***

# `npm install`

The package was axios. 100M weekly downloads. Two months ago.




Sahil Bansal

Software Engineer @SafeDep Inc.

GH, X, Li: @sahilb315

https://sahilbansal.dev

# Agenda

# We've been defending the wrong layer

# mini Shai-Hulud

*self-replicating npm worm · @redhat-cloud-services · Jun 2026*

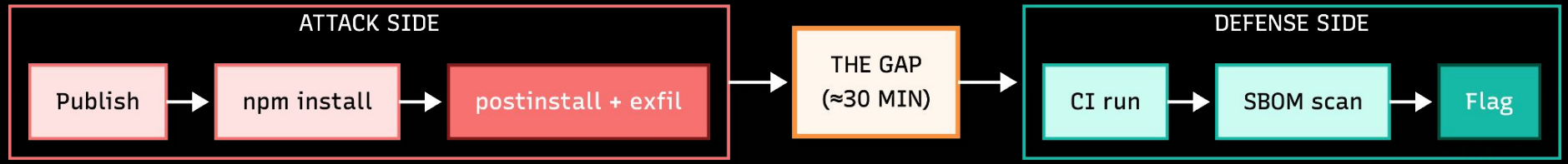
| | | |
|---|---|---|
| 1 | **Account takeover** | Maintainer's npm credentials stolen |
| 2 | **Preinstall hook added** | Fires on every npm install of the package |
| 3 | **Credentials grabbed** | ~/.npmrc · Vault tokens · AWS metadata service |
| 4 | **Persistence attempted** | Sudoers writes on Linux hosts |
| 5 | **Worm spreads** | Injects itself into downstream packages |

# What we've built so far

- SBOMs
- SCA in CI
- Registry detection
- OpenSSF Scorecard
- SLSA provenance
- Dependency review

**All of it activates AFTER the package is on your disk.**

# The supply chain timeline



"The gap is install-time.

That's the front door.

We've been guarding
the back door very well."

# What an attack actually looks like

The attack surface

# Three doors in

### TYPOSQUATTING

You pick the wrong package. lodahs instead of lodash.

### DEPENDENCY CONFUSION

Your tooling picks the wrong package. Internal name, public registry.

### COMPROMISED PACKAGE

The right package, hijacked at the source.
Real axios — now malicious.
Demo next.

*Three doors. One chokepoint: the install.*

DEMO · 01

# What 2 seconds costs you.

```
# your machine

~/demo →
```

```
# attacker terminal  — incoming exfil

[honeypot] listening on http://127.0.0.1:9999
[honeypot] waiting for exfiltration...
```

# Why install-time defense is hard

Challenge 1

# Speed

```
$ brew uninstall pmg
```

*What happens if your security tool adds 3+ seconds to install.*

---

***Performance isn't a feature. It's the price of being run at all.***

github.com/safedep/pmg

Challenge 2

# Compatibility

npm | pnpm | yarn | bun | npx | pnpx | pip | poetry | uv

Build for npm only?

**You miss 8 of the 9 ways developers install code.**

Challenge 3

# Day-zero malware

The axios attack pre-staged 18 hours before


***You need real-time analysis. Not a static blocklist.***

Challenge 4

# Scripts must run

**BLOCK ALL ✗**

Half the ecosystem breaks. node-gyp, native modules, build tools.

**LET ALL ✗**

Postinstall hooks ship malware. You just saw this happen.

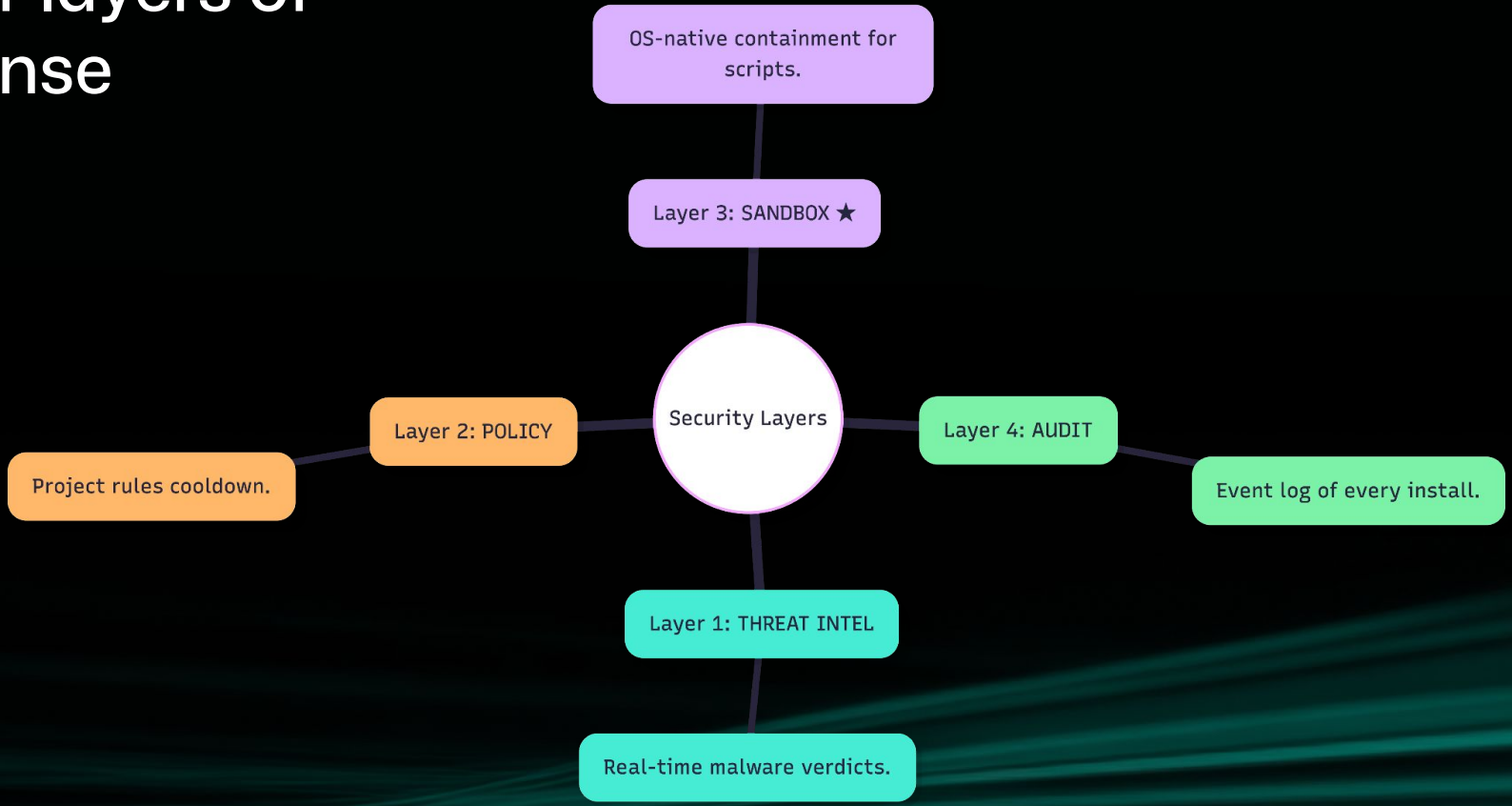
**CONTAIN ✓**

Scripts run, but in a sandbox. Native modules work. Attackers don't.

***Containment, not prohibition.***

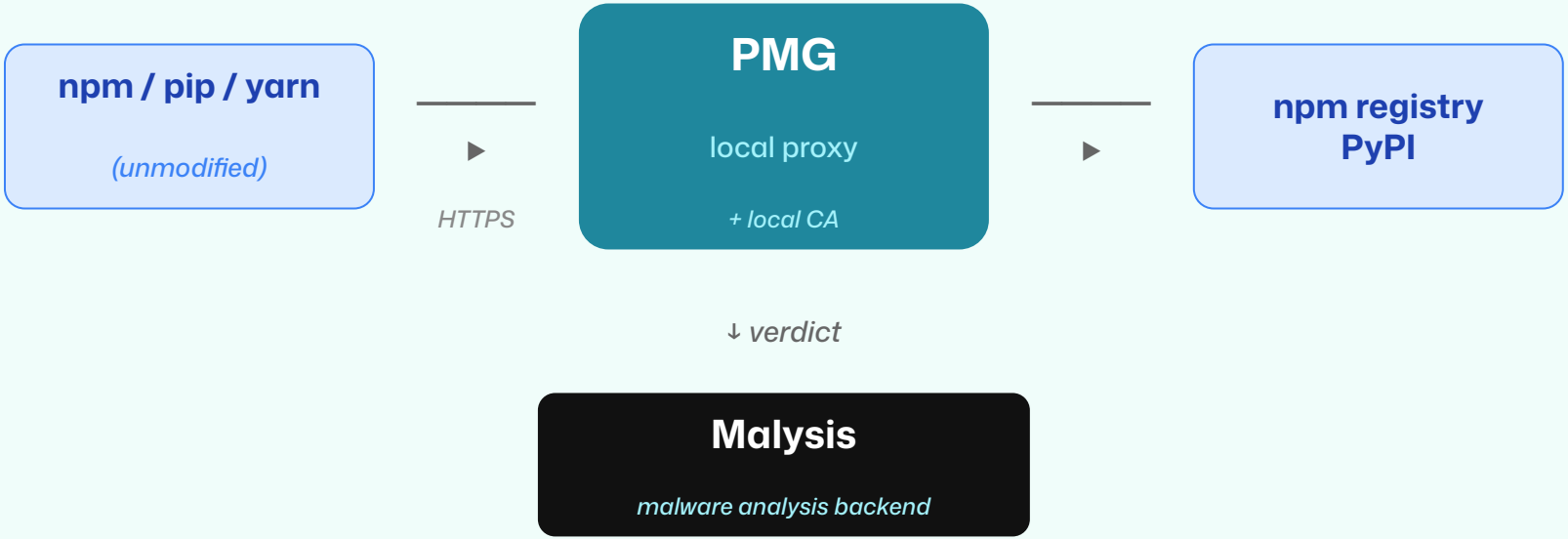
# A different approach

# Four layers of defense



Layer 1

# Threat Intel




***Every package gets a verdict in flight. Direct and transitive.***

github.com/safedep/pmg

Layer 2

# Policy (Dependency Cooldown)

*Wait N days. Threat intel catches up.*


The axios attacker pre-staged plain-crypto-js 18 hours early.

**A 7-day cooldown, which PMG ships today, would have caught it.**

*Sometimes you don't need to know if it's malicious. You just wait.*

Layer 3

# Sandbox

***The script runs. It just cannot read your SSH keys.***

**✗ CANNOT ACCESS**

- ✗ `~/.ssh/id_rsa`
- ✗ `~/.aws/credentials`
- ✗ `~/.npmrc`
- ✗ `~/.bashrc`
- ✗ `cron / launchd`
- ✗ `.git/hooks/`

⟹ *blocked*

**npm-restrictive sandbox**

***postinstall.js runs here***

```
my-app/
├── node_modules/   ✓
├── package.json    ✓
├── src/            ✓
└── ...             ✓
```

*Built on Landlock + seccomp + namespaces (Linux), Seatbelt (macOS). Opt-in today.*

github.com/safedep/pmg

Layer 4

# Audit

*Every install, what, when, who. Forensic trail when threat intel catches up later.*

```
2026-06-07 10:03  install   axios@1.7.7          alice@laptop-32
2026-06-08 11:12  install   lodash@4.17.21       alice@laptop-32
2026-06-09 05:29  install   plain-crypto-js@4.2.1 alice@laptop-32
2026-06-10 14:23  install   react@18.2.0         alice@laptop-32
[ ⚠️ flagged 2 days later ]  plain-crypto-js@4.2.1  → alice exposed
```

***Defense in depth needs memory.***

DEMO · 02

# When npm install says no.

```
~/demo ➜
```

# What's missing and what we're doing

*We know about these. We're working on them.*

| TODAY'S GAP | | WHAT WE'RE SHIPPING |
|---|---|---|
| Backend round-trips per package | ▸ | Local package cache |
| Sandbox is opt-in | ▸ | Default-on sandbox |
| Private registries: limited | ▸ | Better corp + monorepo support |
| Node.js & Python only | ▸ | User-driven ecosystems |

github.com/safedep/pmg

“

SBOMs aren't wrong.

CI scanning isn't wrong.

***They're not sufficient.***

**Install-time matters.**

**Containment is the unlock.”**

github.com/safedep/pmg

*PMG is open source · Apache-2.0*

# Try it on your laptop today.

```
$ brew install safedep/tap/pmg
```

**github.com/safedep/pmg**

Discord active. PRs welcome.

*scan to open repo*

*We'd love help with: Windows sandboxing · more ecosystems · policy ergonomics*

SafeDep

# Thank You


Ship Code.
Not Malware.
SafeDep
github.com/safedep/pmg