



# Protect yourself against supply chain attacks

Rob Bos

DevOps Consultant – Xebia

<https://devopsjournal.io>



<https://myoctocat.com>

# Security issues are ubiquitous

**CNN BUSINESS** Markets Tech Media Calculators Videos

## Massive SolarWinds hack has big businesses on high alert

By Rishi Iyengar, CNN Business  
Published 10:14 AM EST, Sat December 10, 2022

FEDERAL TRADE COMMISSION  
PROTECTING AMERICA'S CONSUMERS

f t m e

### FTC warns companies to remediate Log4j security vulnerability

Log4j is a ubiquitous piece of software used to record activities in a wide range of systems found in consumer-facing products and services. Recently, a serious vulnerability in the popular Java logging package, Log4j (CVE-2021-44228) was disclosed, posing a severe risk to millions of consumer products to enterprise software and web applications. This vulnerability is being widely exploited by a growing set of attackers.

TC  
Join TechCrunch+  
Login  
Search q  
TechCrunch+  
Startups  
Venture

Men

Security

## Microsoft AI researchers accidentally exposed terabytes of internal sensitive data

Carly Page / 3:05 PM GMT+2 • September 18, 2023

Comment

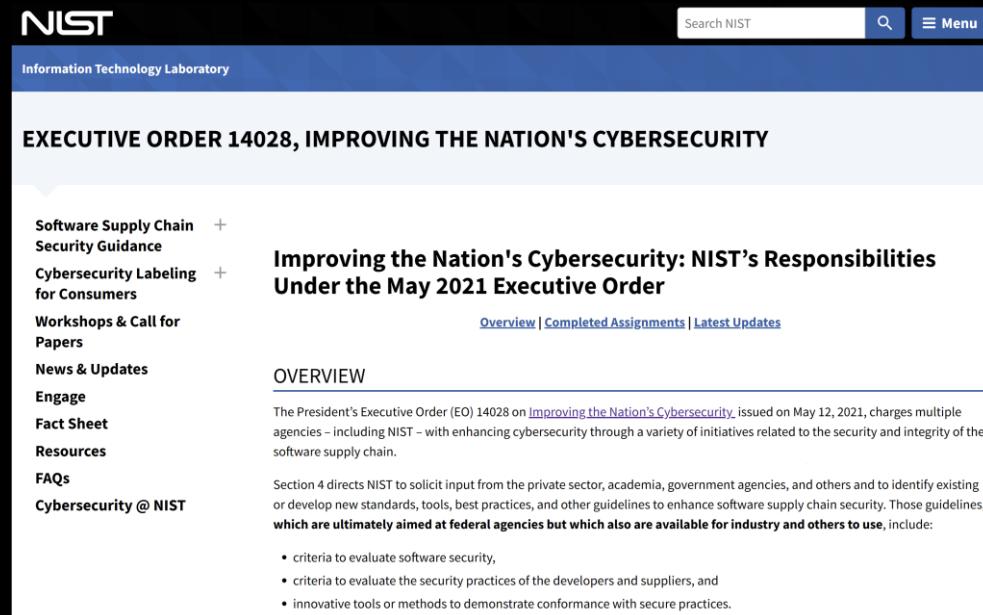


Image Credits: Getty Images

Microsoft AI researchers accidentally exposed tens of terabytes of sensitive data, including private keys and passwords, while publishing a storage bucket of open source training data on GitHub.

# Governmental Regulations

USA, 2021



The screenshot shows the NIST website for Executive Order 14028. The header includes the NIST logo and a search bar. The main title is "EXECUTIVE ORDER 14028, IMPROVING THE NATION'S CYBERSECURITY". On the left, there's a sidebar with links for Software Supply Chain, Security Guidance, Cybersecurity Labeling for Consumers, Workshops & Call for Papers, News & Updates, Engage, Fact Sheet, Resources, FAQs, and Cybersecurity @ NIST. The main content area features a section titled "Improving the Nation's Cybersecurity: NIST's Responsibilities Under the May 2021 Executive Order" with sub-links for Overview, Completed Assignments, and Latest Updates. Below this is an "OVERVIEW" section describing the President's Executive Order (EO) 14028 on improving cybersecurity, issued on May 12, 2021. It states that the EO directs NIST to solicit input from the private sector, academia, government agencies, and others to identify existing or develop new standards, tools, best practices, and other guidelines to enhance software supply chain security. The section also lists criteria for evaluating software security, security practices of developers and suppliers, and innovative tools or methods to demonstrate conformance with secure practices.

EU, 2024 to member states, 24 months



The screenshot shows the European Commission website. The header features the European Commission logo and a language selection for English. The main navigation menu includes Home, Policies, Activities, News, Library, Funding, Calendar, and Consultations. The breadcrumb navigation shows the user is at Home > Policies > The EU Cybersecurity Act. The main content area has a large heading "The EU Cybersecurity Act". Below the heading, a text block states: "The Cybersecurity Act strengthens the EU Agency for cybersecurity (ENISA) and establishes a cybersecurity certification framework for products and services."

# Governmental Regulations

EU 2024: NIS2

Research / Advanced search / The NIS2 Directive: A high common level of cybersecurity in the EU

## The NIS2 Directive: A high common level of cybersecurity in the EU

Briefing – 08-02-2023

The Network and Information Security (NIS) Directive is the first piece of EU-wide legislation on cybersecurity, and its specific aim was to achieve a high common level of cybersecurity across the Member States. While it increased the Member States' cybersecurity capabilities, its implementation proved difficult, resulting in fragmentation at different levels across the internal market. To respond to the growing threats posed with digitalisation and the surge in cyber-attacks, the Commission has submitted a proposal to replace the NIS Directive and thereby strengthen the security requirements, address the security of supply chains, streamline reporting obligations, and introduce more stringent supervisory measures and stricter enforcement requirements, including harmonised sanctions across the EU. The proposed expansion of the scope covered by NIS2, by effectively obliging more entities and sectors to take measures, would assist in increasing the level of cybersecurity in Europe in the longer term. Within the European Parliament, the file was assigned to the Committee on Industry, Research and Energy. The committee adopted its report on 28 October 2021, while the Council agreed its position on 3 December 2021. The co-legislators reached a provisional agreement on the text on 13 May 2022. The political agreement was formally adopted by the Parliament and then the Council in November 2022. It entered into force on 16 January 2023, and Member States now have 21 months, until 17 October 2024, to transpose its measures into national law. Fourth edition. The 'EU Legislation in Progress' briefings are updated at key stages throughout the legislative procedure.

f  
t  
in

EU, 2023 to member states, 24 months

Home | About | Browse by topic | Tools and data | Document Library | Media |

Home > Digital Operational Resilience Act (DORA)

## Digital Operational Resilience Act (DORA)

The [Digital Operational Resilience Act \(DORA\)](#) is a EU regulation that entered into force on 16 January 2023 and will apply as of 17 January 2025.

It aims at strengthening the IT security of financial entities such as banks, insurance companies and investment firms and making sure that the financial sector in Europe is able to stay resilient in the event of a severe operational disruption.

DORA brings harmonisation of the rules relating to operational resilience for the financial sector applying to 20 different types of financial entities and ICT third-party service providers.



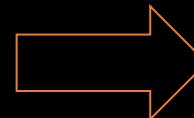
# Topics

- Why:
  - Attack examples
- Protection / Maturity: frameworks
  - OWASP Software Component Verification Standard (SCVS)
  - Supply chain Levels for Software Artifacts (SLSA)

# Supply Chain - Dependencies

- Libraries used by your application:

- Authentication
- Encryption
- Database connections



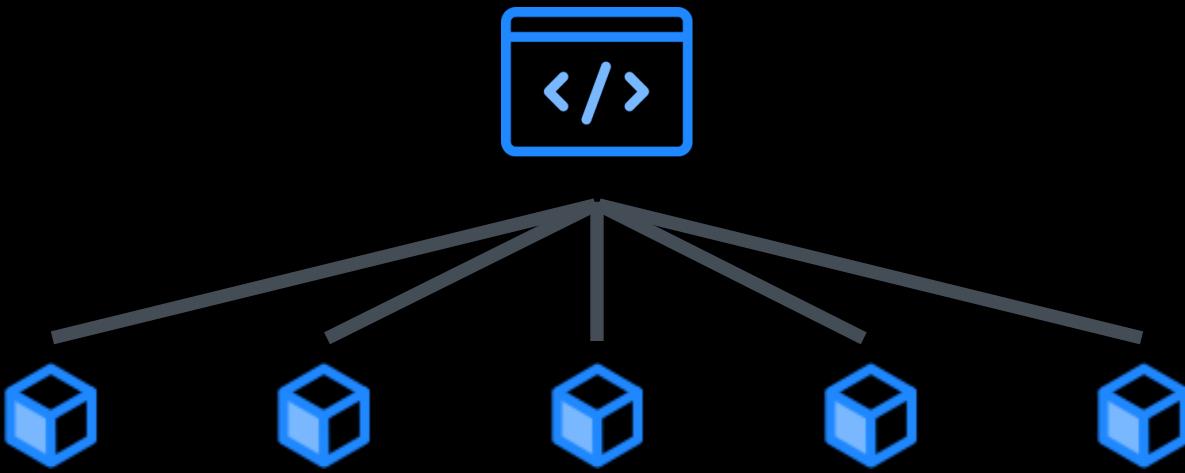
Package managers

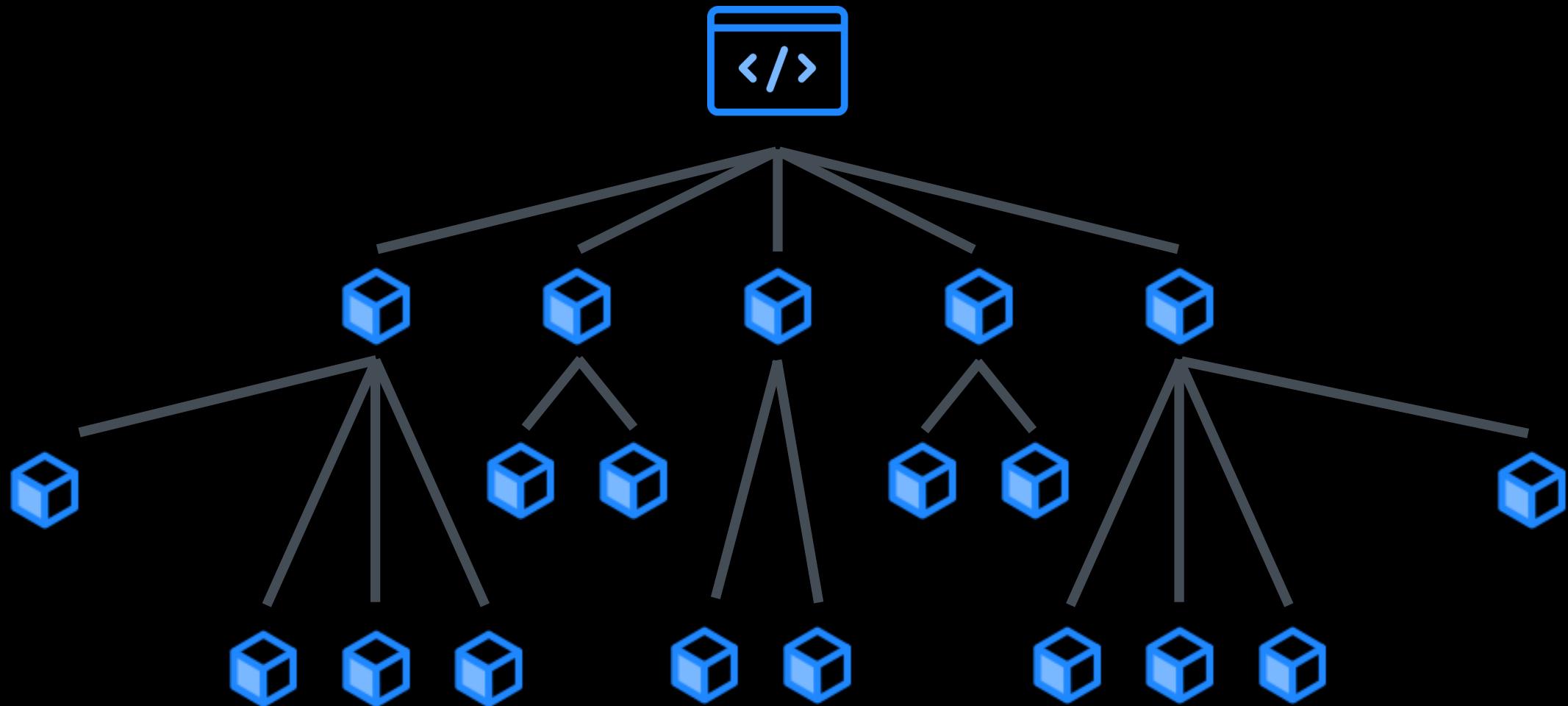
- Tooling used for building your application:

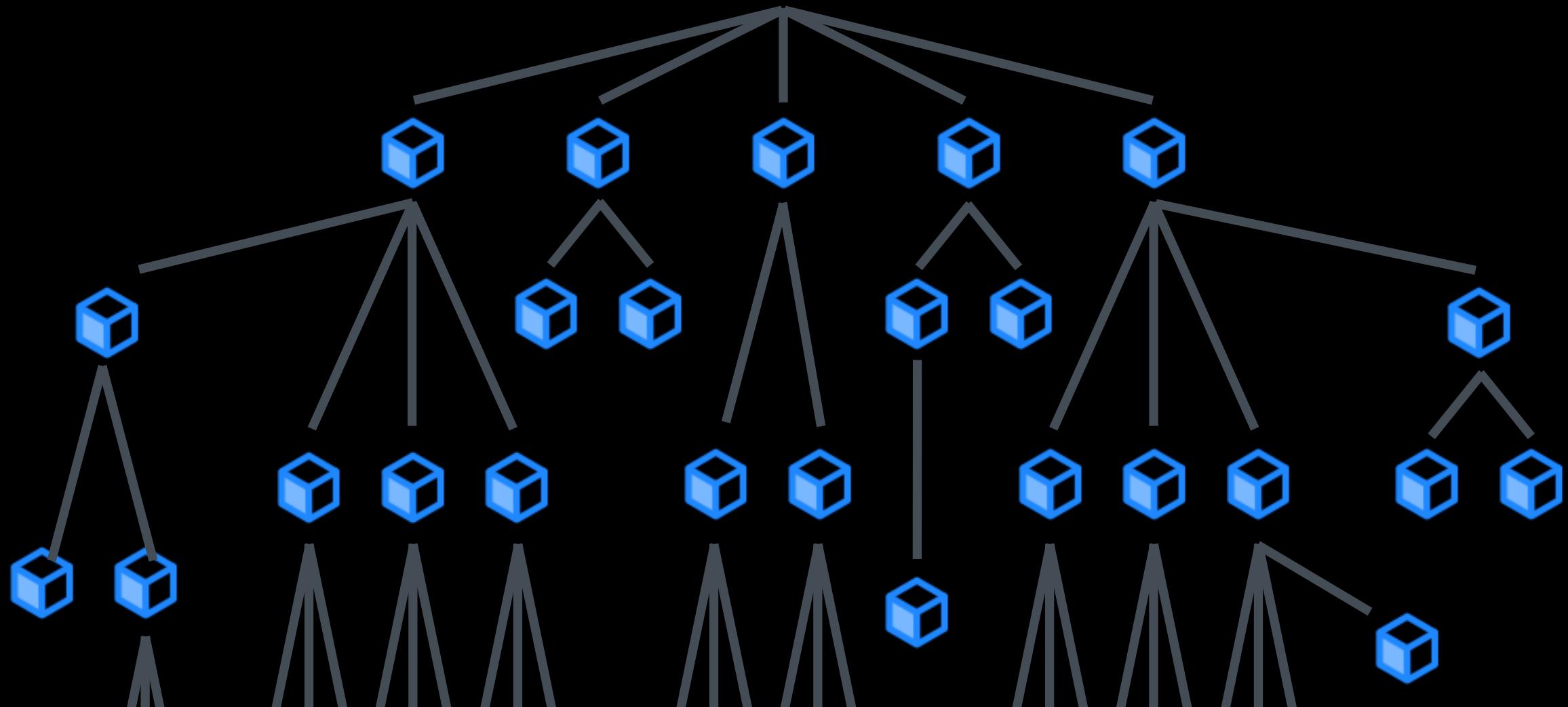
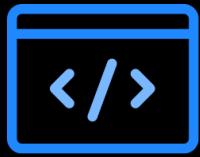
- npm ci
- dotnet build
- pipelines

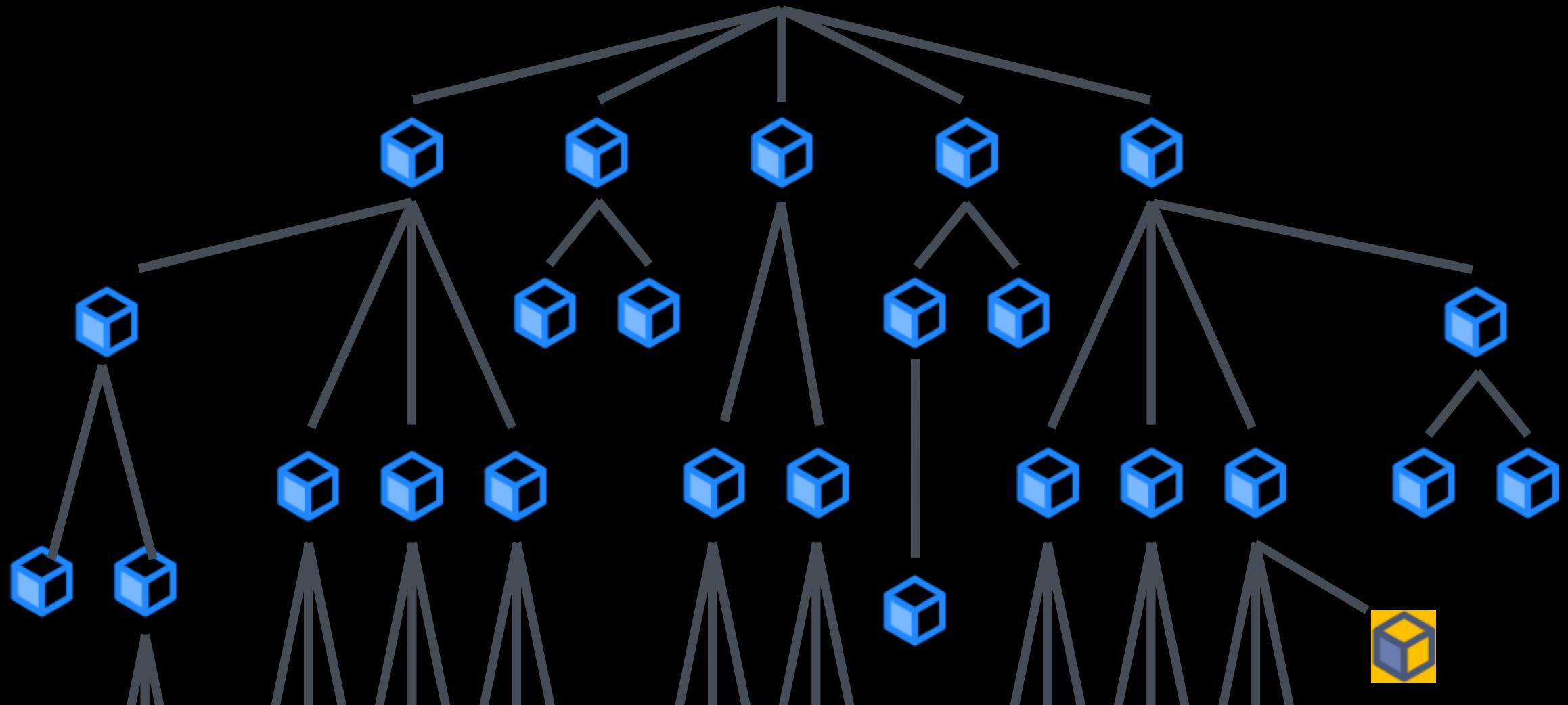
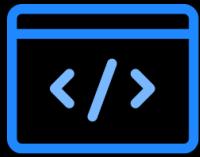


Build tools









# Supply chain depth

- npm cli: <https://github.com/npm/cli/network/dependencies>

**~80-90% of all code in distributed  
applications is Open Source**



# 4 years

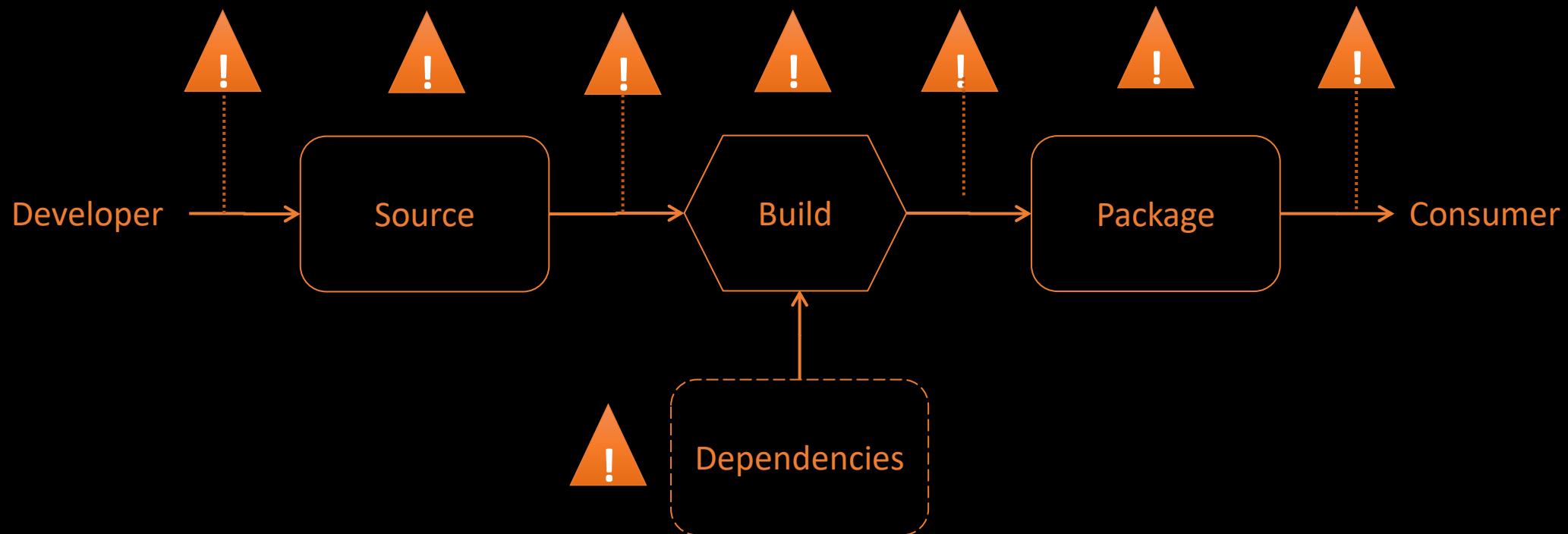
On average, vulnerabilities go undetected for four years before being identified.

Sometimes, even longer than that:  
Log4j was vulnerable for ~7 years

# 180+ days

**Mean time to remediate (MTTR)**  
**Industry norm**

# Attack entries



# Solorigate / SolarWinds – Orion

SolarWinds produces Network monitoring software



# Supply Chain Confusion

- **Typo squatting**
- **Namespace shadowing**
- **Configuration files**
  
- **Pipeline attacks**
- **Pipeline artifact attacks**

# Typo squatting 101

npm install crossenv



Steals all your  
environment variables

npm install cross-env



Normal package

# Add Namespaces

```
npm install @babel/helper-regex
```

More specific, less chance of squatting

Not all publishers use a namespace

# March 2022 – Namespace confusion

`npm install @azure/core-tracing`

vs

`npm install core-tracing`

→ 218 packages squatted

# Timelines

- Attacks spread fast, lots of targets
- @azure namespace attack: 50 downloads per package (x 218 packages!)
- Found within 1-2 days
- Blocked by npm after noticing →
- Damage is already done by then

The screenshot shows a package page for 'core-tracing' version 0.0.1-security. The page indicates the package was published 6 days ago and is public. It features three main buttons: 'Readme' (highlighted in yellow), 'Explore' (in red), and 'Dependencies' (in purple). A large heading 'Security holding package' is displayed, followed by a note explaining the package was removed from the registry by the npm security team due to malicious code. A link to the advisories page is provided at the bottom.

core-tracing

0.0.1-security • Public • Published 6 days ago

Readme Explore BETA

0 Dependencies

## Security holding package

This package contained malicious code and was removed from the registry by the npm security team. A placeholder was published to ensure users are not affected in the future.

Please refer to [www.npmjs.com/advisories?search=core-tracing](http://www.npmjs.com/advisories?search=core-tracing) for more information.

# Typo squatting 101

And this is only getting worse!

## The danger of .zip domains

This downloads postgres v15

`https://github.com/postgres/postgres/tags/v15.zip`

hostname

This resolves to v15.zip domain

`https://github.com/postgres/postgres/tags/@v15.zip`

userinfo

hostname

# Protect yourself

- Software Composition Analysis (SCA) is the starting point
- Use SAST or DAST (shift left or you are too late!)
- Package manager scanners:
  - WhiteSource
  - BlackDuck
  - JFrog Artifactory + Xray
  - Snyk.io
  - GitHub Dependabot + Vulnerability alerts



Checks package + version against  
CVE databases

AFTER THE FACT

# Protect yourself

npm audit

- Gets a list of the dependencies and posts that to:

<https://registry.npmjs.org/-/npm/v1/security/advisories/bulk>

- Might send out private data!
- Any packages without a version field will be ignored!

AFTER THE FACT

# npm audit - results

```
node_modules/url-parse

ws 5.0.0 - 5.2.2 || 6.0.0 - 6.2.1
Severity: moderate
ReDoS in Sec-WebSocket-Protocol header - https://github.com/advisories/GHSA-6fc8-4gx4-v693
ReDoS in Sec-WebSocket-Protocol header - https://github.com/advisories/GHSA-6fc8-4gx4-v693
fix available via `npm audit fix`
node_modules/jest-environment-jsdom-fourteen/node_modules/ws
node_modules/webpack-dev-server/node_modules/ws
node_modules/ws

57 vulnerabilities (25 moderate, 31 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force
```

# npm install -g npq

- Wrapper from snyk.io
- Alias it to overwrite npm commands

```
→ npm install amp-html
  ✓ Checking package maturity
  ✗ Identifying package author...
  ✗ Checking package download popularity
  ✓ Checking availability of a LICENSE
  ✓ Checking availability of a README
  ✓ Identifying package repository...
  ✓ Checking package for pre/post install scripts
  ✗ Checking for known vulnerabilities

Detected possible issues with the following packages:
[amp-html]
  - the package description has no e-mail associated with author(s). Proceed with care.
  - detected a low download-count package (downloads last month < 20)
  - 1 vulnerabilitie(s) found: https://snyk.io/vuln/npm:amp-html

? Would you like to continue installing package(s)? (y/N) █
```

# .npmrc - misconfiguration

registry=https://registry.npmjs.org/

@myscope:registry=https://mycustomregistry.example.org

All your private packages will now get pulled from npmjs.org!

# dependency-review-action

dependency-review summary

## Dependency Review

The following issues were found:

- ✗ 1 vulnerable package(s)
- ✅ 0 package(s) with incompatible licenses
- ✅ 0 package(s) with invalid SPDX license definitions
- ⚠ 2 package(s) with unknown licenses.

See the Details below.

### Vulnerabilities

*Cargo.toml*

Name	Version	Vulnerability	Severity
Simple-Wayland-HotKey-Daemon	$\geq 1.1.4, < 1.2.0$	<a href="#">Insecure Temporary File in SWHKD</a>	critical
		<a href="#">Insecure temporary file usage in SWHKD</a>	critical

# Topics

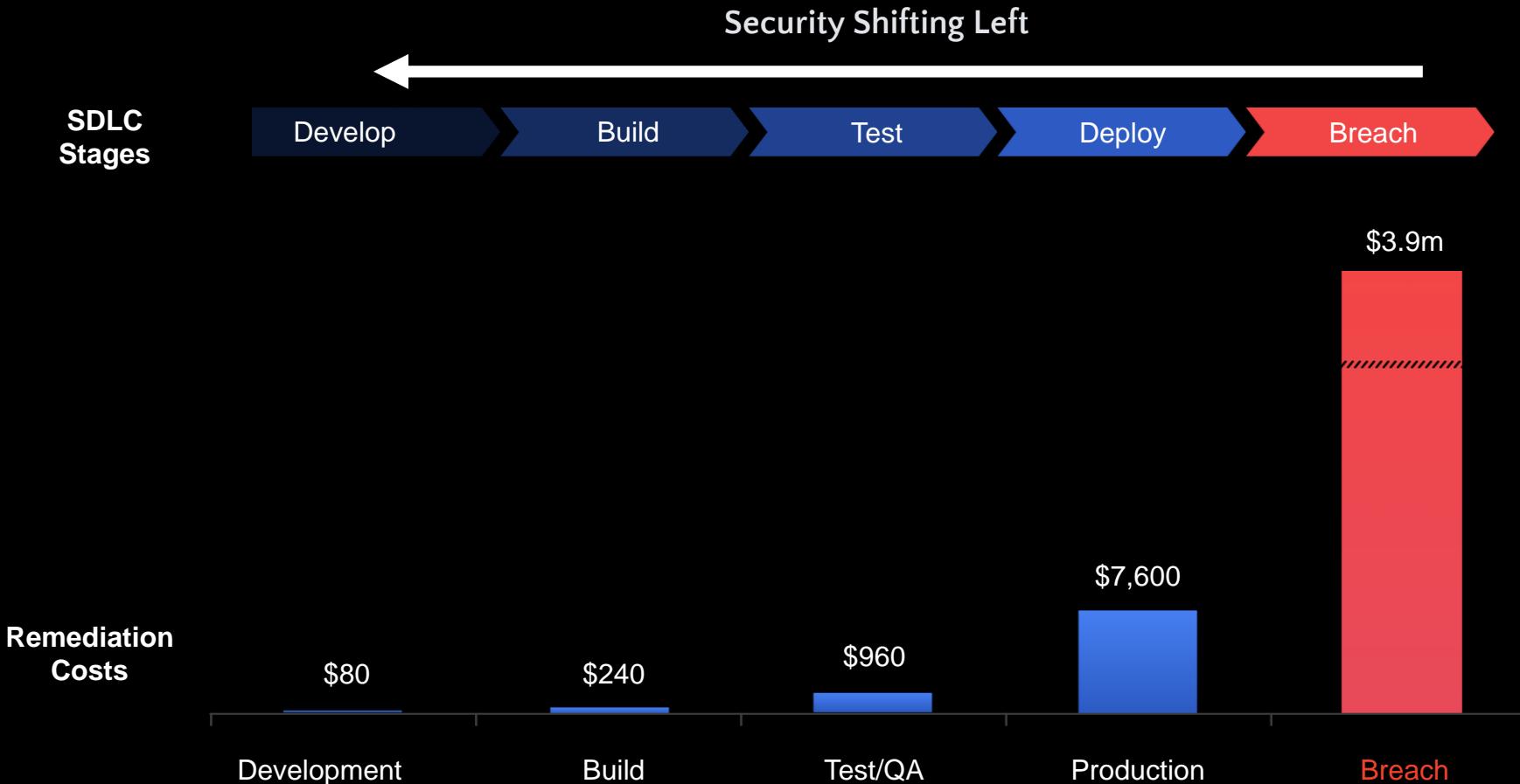
- Why:
  - Attack examples
- Protection / Maturity: frameworks
  - OWASP Software Component Verification Standard (**SCVS**)
  - Supply chain Levels for Software Artifacts (**SLSA**)



# Mistakes happen

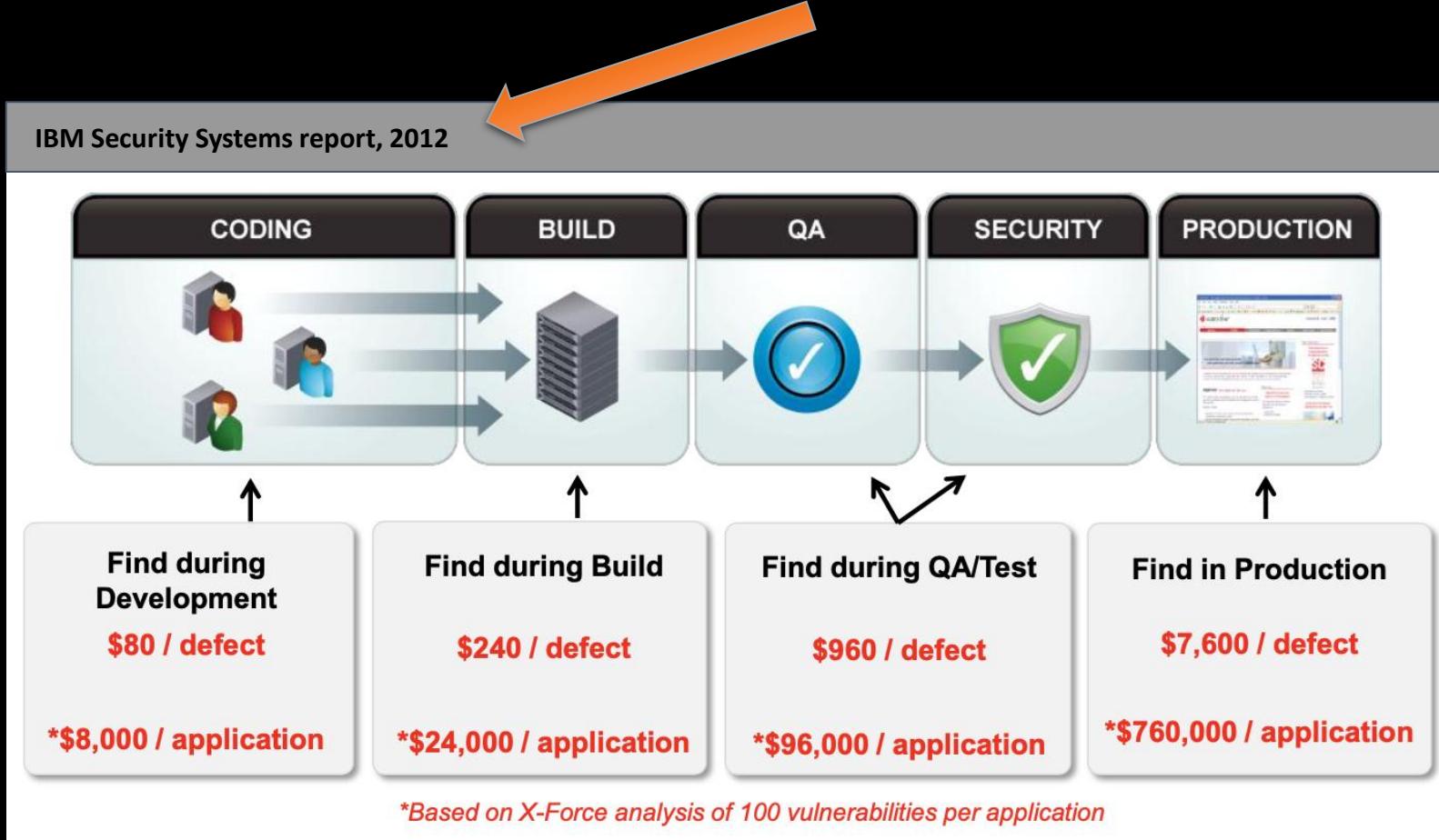
Some of those become a security risk!

# Everyone wants to shift security left...



Source: Ponemon Institute Cost of a Data Breach 2020

... but the industry has been trying  
to shift left for at least a decade



# Frameworks



**OWASP**  
Software Component Verification Standard  
v1 since 2020: <https://xpir.it/SCVS>



The Linux Foundation  
Supply chain Levels for Software Artifacts  
v1.0: April 2023: <https://slsa.dev/>

# OWASP SCVS: Software Component Verification Standard

- Assessment of your software components and how they came to be



Packages/Source



Pipelines

# OWASP SCVS

	L1	L2	L3	L4
V1 – Inventory				
V2 – Software Bill of Materials (SBOM)				
V3 – Build Environment				
V4 – Package Management				
V5 – Component Analysis				
V6 – Pedigree and Provenance				

# OWASP SCVS – V1 Inventory

All direct and transitive components and their versions are known at completion of a build

Package managers are used to manage all third-party binary components

Software bill of materials continuously maintained and current for all systems

Software bill of materials are required for new procurements

The component type is known throughout inventory

The component function is known throughout inventory

# V1 - Inventory

## Software Composition Analysis

- GitHub Dependabot
- Black Duck
- Mend (form. WhiteSource) Bolt
- Snyk
- Jfrog Xray

Dependabot Example: <https://github.com/devops-actions/.github>

# OWASP SCVS – V2 Software Bill of Materials

- SBOM creation is automated and reproducible
- SBOM has been signed by publisher, supplier, or certifying authority
- SBOM signature verification exists and is performed
- SBOM is analyzed for risk

# V2 Software Bill of Materials

- Multiple standards for SBOM formats:
  - SPDX (Software Package Data Exchange) – Linux Foundation
    - Focuses on license information
    - ISO Standard
  - CycloneDX – OWASP
    - Focuses on vulnerabilities and security
- Example SBOM creation with GitHub Actions & CycloneDX:  
[AccelerateDevOps/sbom.yml at main · wulfland/AccelerateDevOps · GitHub](https://github.com/wulfland/AccelerateDevOps/blob/main/.github/workflows/sbom.yml)

# V2 Software Bill of Materials - demo

The screenshot shows a GitHub Actions workflow run for a repository named 'wulfland / AccelerateDevOps'. The workflow is titled 'Generate SBOM' and has run number #6. The 'Actions' tab is selected in the navigation bar.

The workflow file for this run is .github/workflows/sbom.yml at fe37a09. The code is as follows:

```
1  name: Generate SBOM
2
3  on: [workflow_dispatch]
4
5  jobs:
6    build:
7      runs-on: ubuntu-latest
8
9    steps:
10   - name: Checkout Code
11     uses: actions/checkout@v2
12
13   - name: CycloneDX .NET Generate SBOM
14     uses: CycloneDX/gh-dotnet-generate-sbom@v1.0.1
15     with:
16       path: ./ch9_release/src/Tailwind.Traders.Web.sln
17       github-bearer-token: ${{ secrets.GITHUB_TOKEN }}
18
19   - name: Upload a Build Artifact
20     uses: actions/upload-artifact@v2.3.1
21     with:
22       path: bom.xml
```

An orange arrow points to the step where the CycloneDX .NET Generate SBOM action is used.

# V2 Software Bill of Materials - demo

```
bom.xml  X

C: > Users > RobBos > AppData > Local > Temp > Temp1_artifact.zip > bom.xml

1  <?xml version="1.0" encoding="utf-8"?>
2  <bom xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" serialNumber=
3    <metadata>
4      <tools>
5        <tool>
6          <vendor>CycloneDX</vendor>
7          <name>CycloneDX module for .NET</name>
8          <version>2.3.0.0</version>
9        </tool>
10       </tools>
11       <component type="application" bom-ref="Tailwind.Traders.Web@0.0.0">
12         <name>Tailwind.Traders.Web</name>
13         <version>0.0.0</version>
14       </component>
15     </metadata>
16     <components>
17       <component type="library" bom-ref="pkg:nuget/Azure.Core@1.8.1">
18         <publisher>Microsoft</publisher>
19         <name>Azure.Core</name>
20         <version>1.8.1</version>
21         <description>This is the implementation of the Azure Client Pipeline</description>
22         <scope>required</scope>
23         <hashes>
24           <hash alg="SHA-512">87135CD530138F27E7C52BA23FB91CE37DE7AE0A016E08D4A5ABF33BD80B0D168996E3A437378B8BDC16667E2
25         </hashes>
26         <licenses>
27           <license>
28             <id>MIT</id>
29           </license>
30         </licenses>
31         <copyright>© Microsoft Corporation. All rights reserved.</copyright>
32         <purl>pkg:nuget/Azure.Core@1.8.1</purl>
33         <externalReferences>
34           <reference type="website">
35             <url>https://github.com/Azure/azure-sdk-for-net/blob/Azure.Core_1.8.1/sdk/core/Azure.Core/README.md</url>
36           </reference>
```

# OWASP SCVS V3 – Build Environment

- Application build pipeline may only perform builds of source code maintained in version control systems
- Application build pipeline prohibits alteration of DNS and network settings during build
- Application build pipeline prohibits alteration of certificate trust stores
- Application build pipeline enforces authentication and defaults to deny
- Application build pipeline enforces authorization and defaults to deny
- Application build pipeline requires separation of concerns for the modification of system settings
- Application build pipeline maintains a verifiable audit log of all system changes
- Application build pipeline maintains a verifiable audit log of all build job changes
- Application build pipeline has required maintenance cadence where the entire stack is updated, patched, and re-certified for use
- Compilers, version control clients, development utilities, and software development kits are analyzed and monitored for tampering, trojans, or malicious code
- All build-time manipulations to source or binaries are known and well defined
- Checksums of all first-party and third-party components are documented for every build
- Checksums of all components are accessible and delivered out-of-band whenever those components are packaged or distributed
- Unused direct and transitive components have been identified
- Unused direct and transitive components have been removed from the application

# OWASP SCVS V3 – Build Environment

- Application build pipeline maintains a verifiable audit log of:
  - build job changes
  - system changes
- The entire build stack is updated, patched, and re-certified for use
- Everything is analyzed and monitored for tampering, trojans, or malicious code
  - Compilers
  - Version control clients
  - Development utilities
  - Software development kits

# V3 – Build Environment

- Application build pipeline maintains a verifiable audit log of:
  - build job changes
  - system changes
- Example in GitHub Actions: end-to-end traceability:
  - Workflow changes in commits
  - Execution environment in the logs
    - <https://github.com/devops-actions/issue-comment-tag/blob/main/.github/workflows/testing.yml>

# V3 – Build Environment - demo

The screenshot shows a GitHub Actions build log for a pull request titled "Bump node-fetch from 2.6.6 to 2.6.7 (#14) Build the action #57". The build status is green, indicating success. The log output is displayed in a monospaced font, showing the setup of the build environment. An orange arrow points to the line where the virtual environment is being set up, specifically highlighting the URL for the Ubuntu 20.04 LTS image.

```
1 Current runner version: '2.289.1'
2 ▼Operating System
3   Ubuntu
4   20.04.4
5   LTS
6 ▼Virtual Environment
7   Environment: ubuntu-20.04
8   Version: 20220227.1
9   Included Software: https://github.com/actions/virtual-environments/blob/ubuntu20/20220227.1/images/linux/Ubuntu2004-Readme.md
10  Image Release: https://github.com/actions/virtual-environments/releases/tag/ubuntu20%2F20220227.1
11 ▼Virtual Environment Provisioner
12   1.0.0-main-20220307-1
13 ▶GITHUB_TOKEN Permissions
14 Secret source: Actions
15 Prepare workflow directory
16 Prepare all required actions
17 Getting action download info
18 Download action repository 'actions/checkout@v2' (SHA:ec3a7ce113134d7a93b817d10a8272cb61118579)
```

# V3 – Build Environment - demo

<https://github.com/actions/virtual-environments/blob/ubuntu20/20220227.1/images/linux/Ubuntu2004-Readme.md>

The screenshot shows a GitHub repository page for `actions/virtual-environments`. The URL in the address bar is <https://github.com/actions/virtual-environments/blob/ubuntu20/20220227.1/images/linux/Ubuntu2004-Readme.md>. The page displays the `Ubuntu2004-Readme.md` file. The commit history shows a single commit from user `459680` on Feb 28, 2022, updating the README file for the `ubuntu20` version `20220227.1`. The file content includes an "Announcements" section with a link to an issue about memory allocation in static TLS blocks, and a "Ubuntu 20.04.4 LTS" section listing the Linux kernel version (`5.11.0-1028-azure`) and image version (`20220227.1`). The "Installed Software" section lists the installed language and runtime, including `Bash 5.0.17(1)-release` and `Clang 10.0.0, 11.0.0, 12.0.0`.

Ubuntu 20.04.4 LTS

- Linux kernel version: 5.11.0-1028-azure
- Image Version: 20220227.1

Installed Software

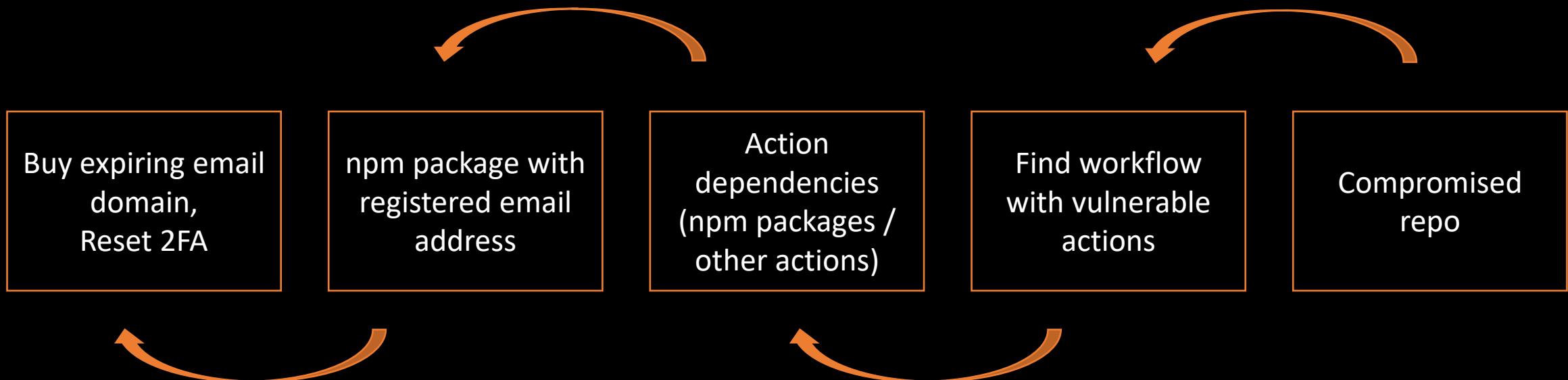
Language and Runtime

- Bash 5.0.17(1)-release
- Clang 10.0.0, 11.0.0, 12.0.0

# What happens during build?

- GitHub Actions: runner will download the actions
- Actions do their thing(s)?
  - <https://xpir.it/universe-2021>
- Harden runner: [link](#)

# GitHub Actions attacks



# OWASP SCVS V4 – Package management

- Package repository components have been published with multi-factor authentication
- Package repository notifies publishers & users of security issues
- Package repository requires code signing to publish packages to production repositories
- Package manager verifies the integrity of packages when they are retrieved:
  - From remote repository
  - From file system

# OWASP SCVS V4 – Package management

- Package repository components have been published with multi-factor authentication – npm improvements with 2FA
- Package repository notifies publishers & users of security issues
  - Dependabot does this
- Package repository requires code signing to publish packages to production repositories
  - Most package managers do not have support
  - npm focusses on 2FA
  - NuGet, Maven / Gradle / Ant (uploads through Maven central), RubyGems

# OWASP SCVS V5 - Component Analysis

- Component is analyzed using linters and/or static analysis tools
- Linting and/or static analysis is performed with every upgrade
- An automated process for:
  - Identifying all publicly disclosed vulnerabilities is used
  - Identifying confirmed dataflow exploitability is used
  - For identifying end-of-life / end-of-support components is used

# V5 - Component Analysis

- Component is analyzed using linters and/or static analysis tools
  - Trigger workflow on push
- Linting and/or static analysis is performed with every upgrade
  - Branch protection rules
- An automated process for:
  - Identifying all publicly disclosed vulnerabilities is used: Dependabot
  - Identifying confirmed dataflow exploitability is used: SAST / DAST
  - For identifying end-of-life / end-of-support components is used: ?

# OWASP SCVS V6 - Pedigree and Provenance

- Point of origin is verifiable for components
- Chain of custody is auditable for components

# V6 - Pedigree and Provenance

- Point of origin is verifiable for components
  - Dependabot dependency graph
- Chain of custody is auditable for components
  - Where did the component came from:
    - Pipeline link
    - Commit history

Example: <https://github.com/devops-actions/issue-comment-tag/releases>

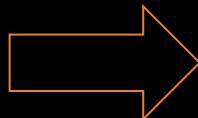
# V6 - Pedigree and Provenance

- Point of origin is verifiable for components
  - Cosign + sigstore = verification options for containers
- Chain of custody is auditable for components
  - SLSA is doing this for other binaries

# Wrap up: Supply Chain - Dependencies

- Libraries used by your application:

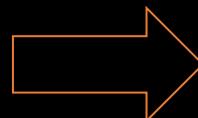
- Authentication
- Encryption
- Database connections



Package managers

- Tooling used for building your application:

- npm ci
- dotnet build
- pipelines



Build tools

# Wrap up – topics:

- Why:
  - Attack examples
- Protection / Maturity: frameworks
  - OWASP Software Component Verification Standard (**SCVS**)
  - Supply chain Levels for Software Artifacts (**SLSA**)



# Rob Bos

**DevOps Consultant | GitHub  
Trainer @ Xebia**

Microsoft MVP + MCT | GitHub Star

GitHub Accredited Trainer

LinkedIn Learning Instructor (GitHub subjects)

Book author: “GitHub Actions in Action”

Blogger: <https://devopsjournal.io>





# Protect yourself against supply chain attacks

Rob Bos

DevOps Consultant – Xebia

<https://devopsjournal.io>



<https://myoctocat.com>