

# W>

# Securing your application software supply-chain

# Niels Tanis

# VERACODE

0101  
0101

# Who am I?

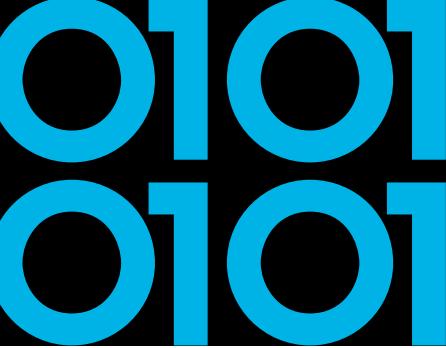
- Niels Tanis
- Senior Principal Security Researcher @ Veracode
  - Background .NET Development, Pentesting/ethical hacking, and software security consultancy
  - ISC<sup>2</sup> CSSLP
  - Research on static analysis for .NET apps



# Securing your application software supply-chain

0101  
0101





# Agenda

- Definition Software Supply-Chain
- Securing the Software Supply-Chain
  - Developer & Source
  - 3<sup>rd</sup> Party Libraries
  - Build & Release
- Conclusion and Q&A

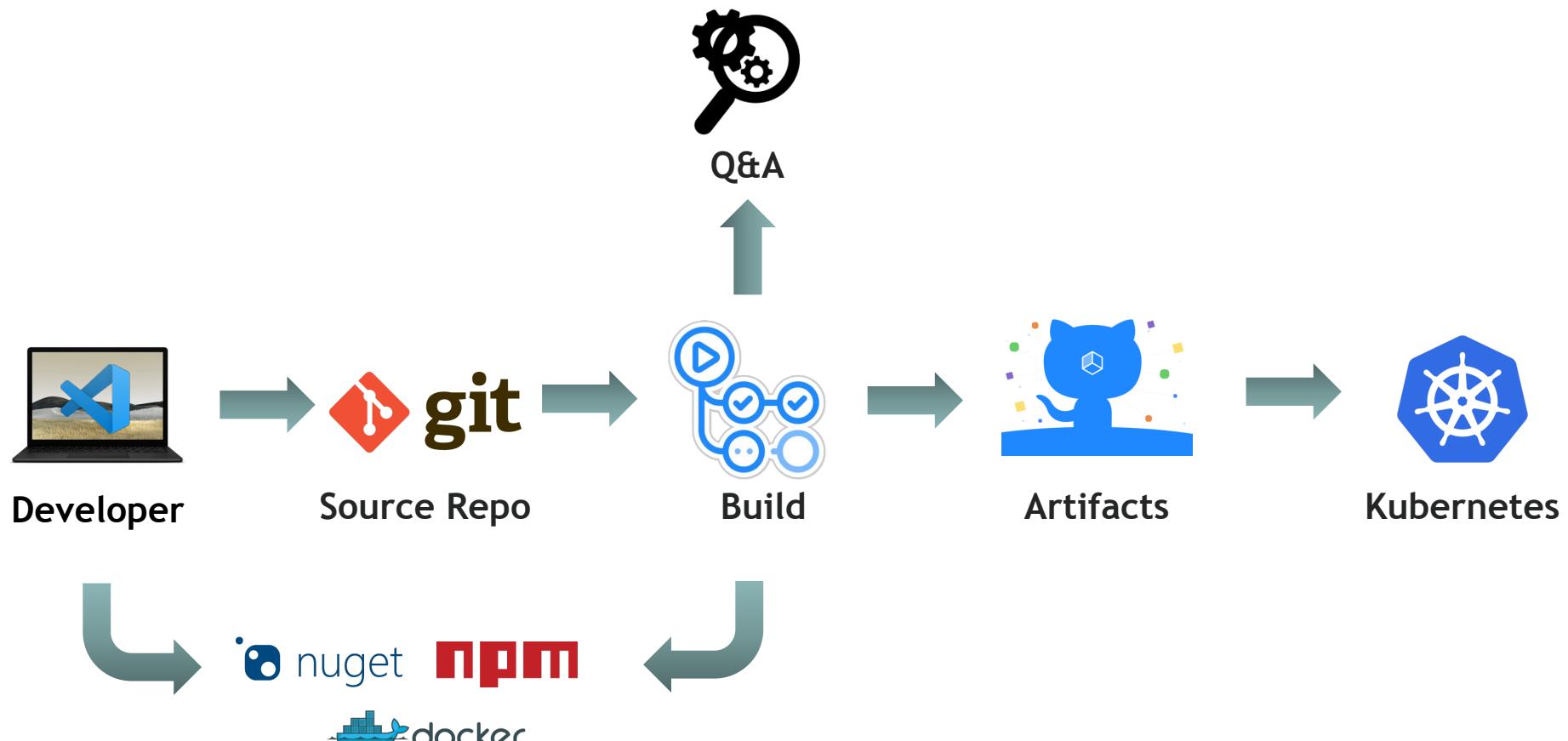
0101  
0101

# What is a Supply Chain?



# Software Supply Chain

0101  
0101



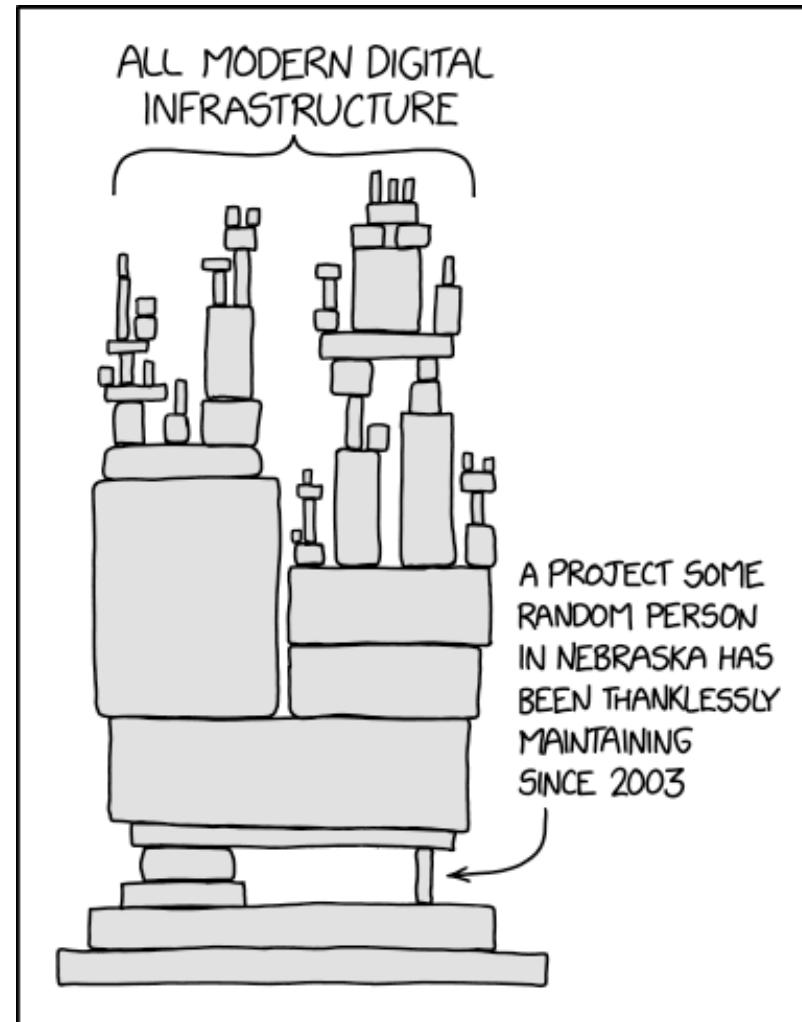
0101  
0101

# SolarWinds SunSpot

A screenshot of a web browser window showing a blog post from CrowdStrike. The title of the post is "SUNSPOT: An Implant in the Build Process". The post is dated January 11, 2021, and is authored by the CrowdStrike Intelligence Team under the Research & Threat Intel category. The background of the page features a large, stylized graphic of a sun with rays emanating from it.

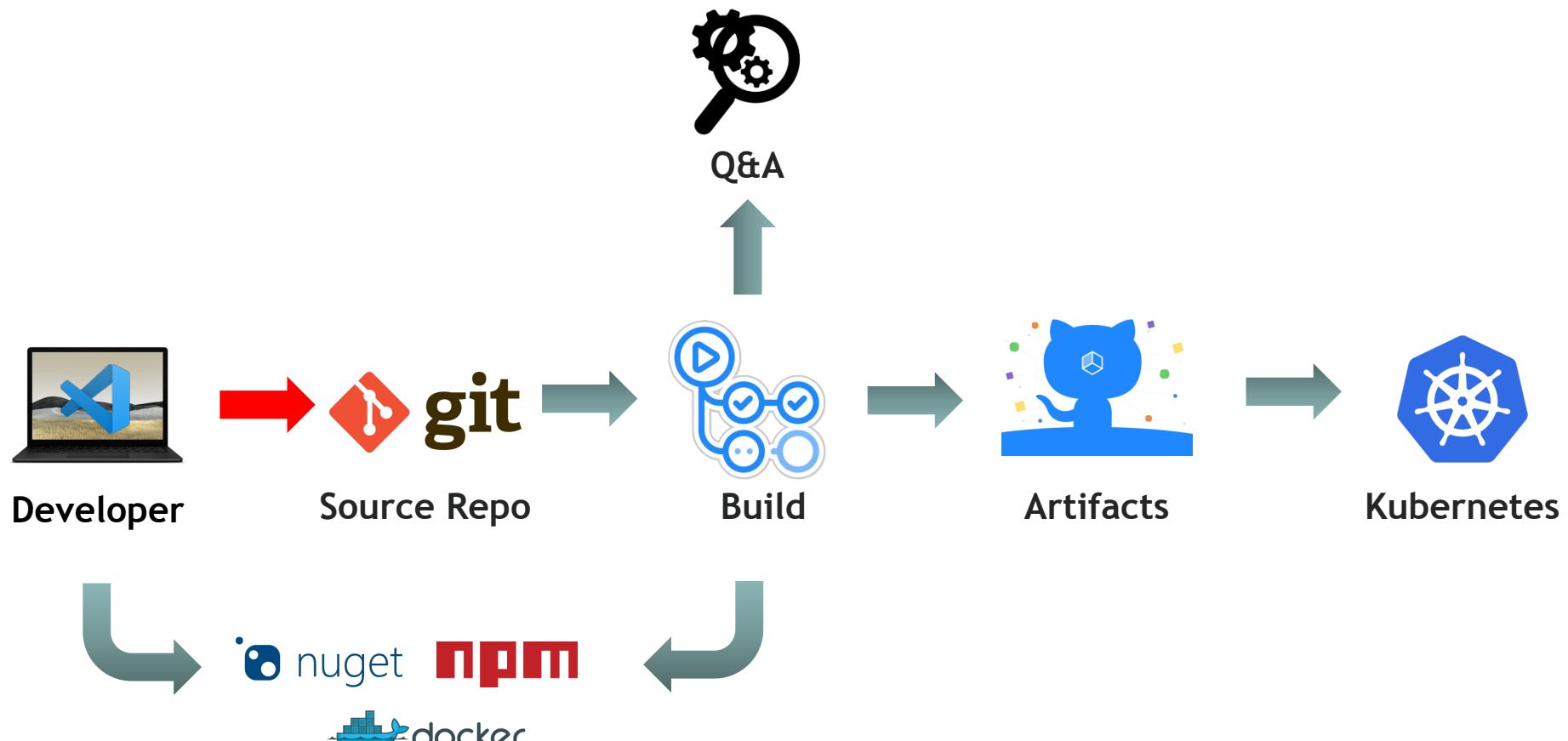
0101  
0101

# XKDC 2347 - Dependency



# Software Supply Chain

0101  
0101



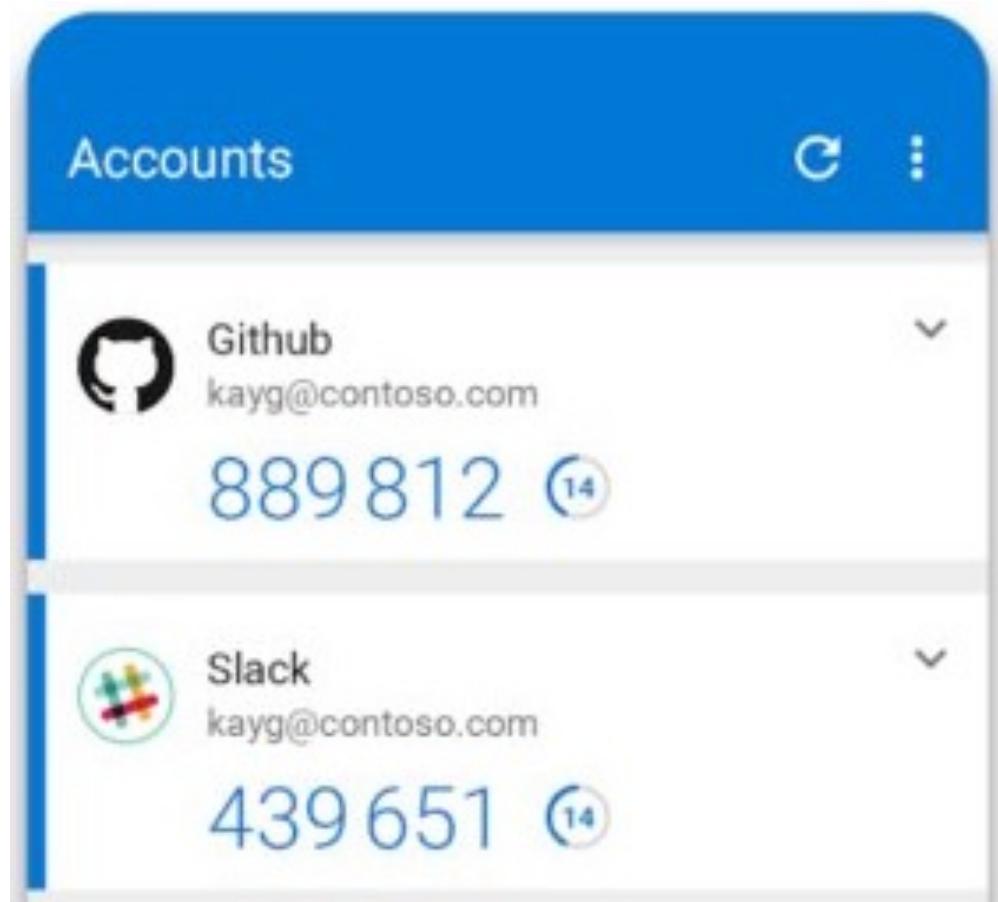
0101  
0101

# GitHub account

The screenshot shows a web browser window displaying a ZDNet article. The URL in the address bar is <https://www.zdnet.com/article/canonical-gith...>. The page title is "Canonical GitHub account hacked, Ubuntu source code safe". Below the title, a sub-headline reads "Ubuntu source code appears to be safe; however Canonical is investigating.". The author is Catalin Cimpanu, and the date is July 7, 2019. The topic is Security. A small image of a person's face is next to the author's name. Below the headline, there is a screenshot of the Canonical Ltd GitHub profile page, showing repositories like "CAN\_GOT\_HAXXD\_10". To the right of the main article, there is a sidebar with a link to "Why everyone should have this cheap security tool". The ZDNet navigation bar at the top includes links for AFRICA, UK, ITALY, SPAIN, MORE, EDITION: EU, NEWSLETTERS, ALL WRITERS, and a user icon.

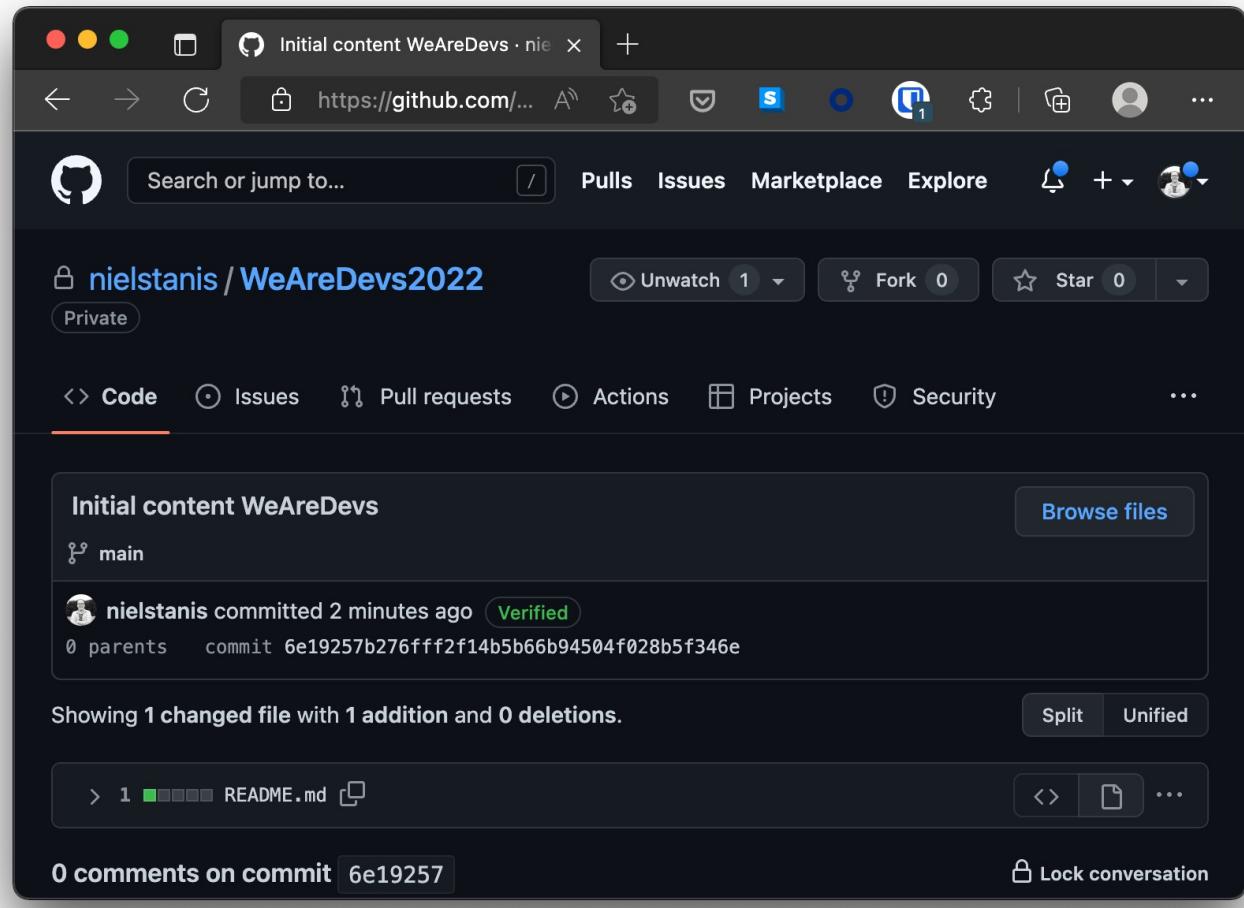
0101  
0101

# Use MFA on source-repository

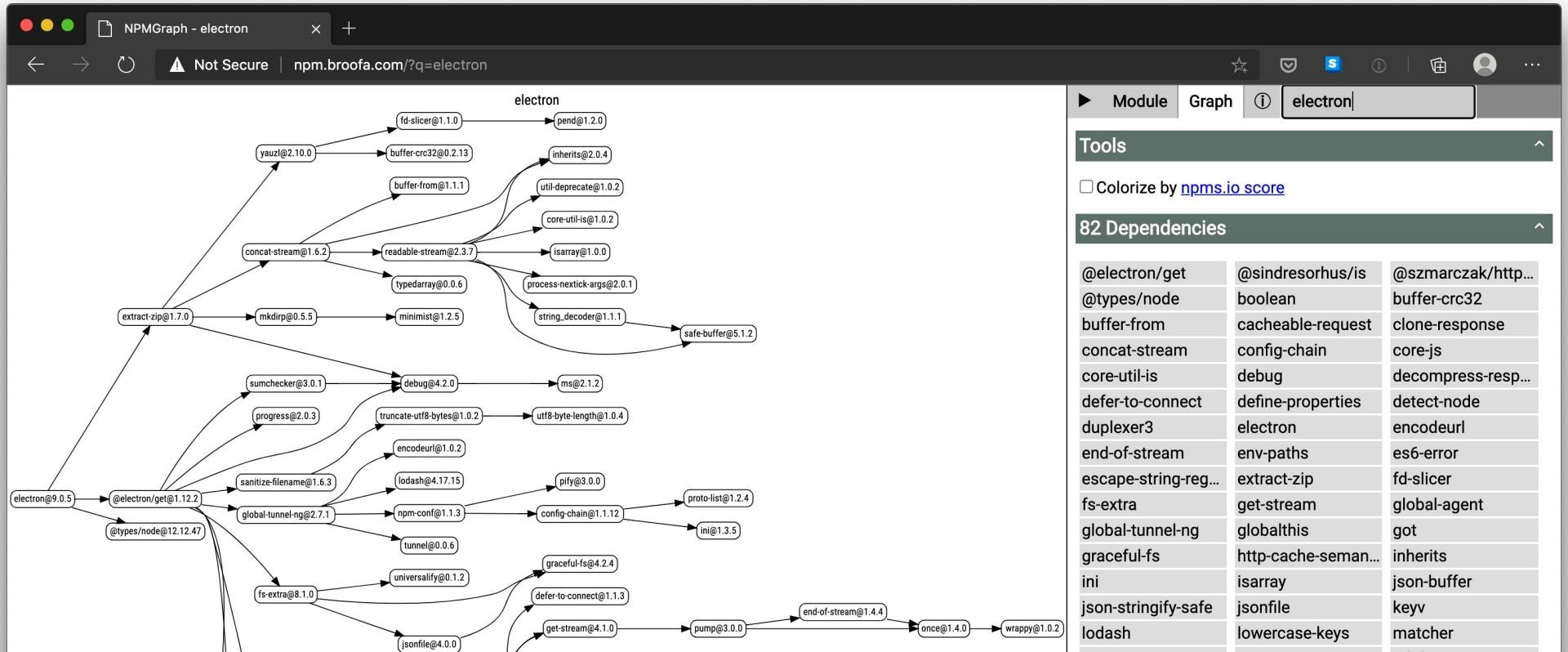
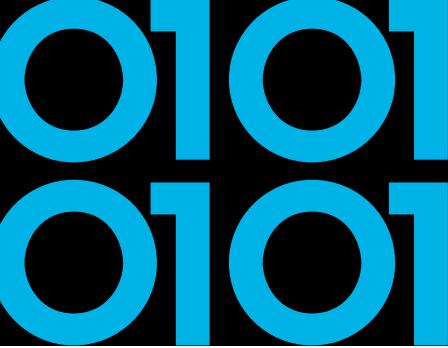


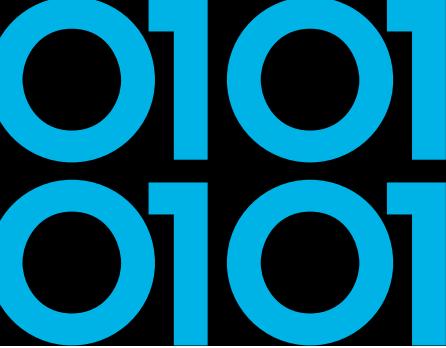
0101  
0101

# GIT Commit Signing

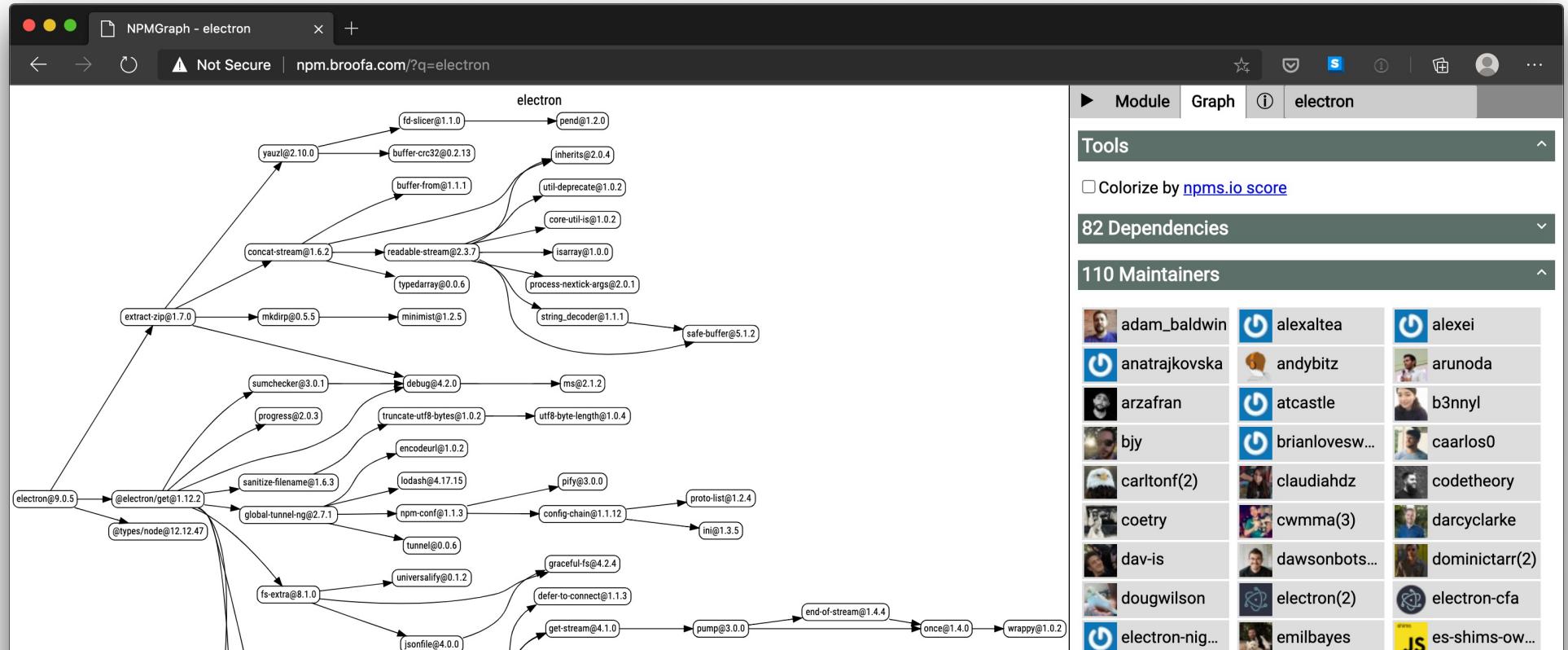


# Visual Studio Code





# Visual Studio Code



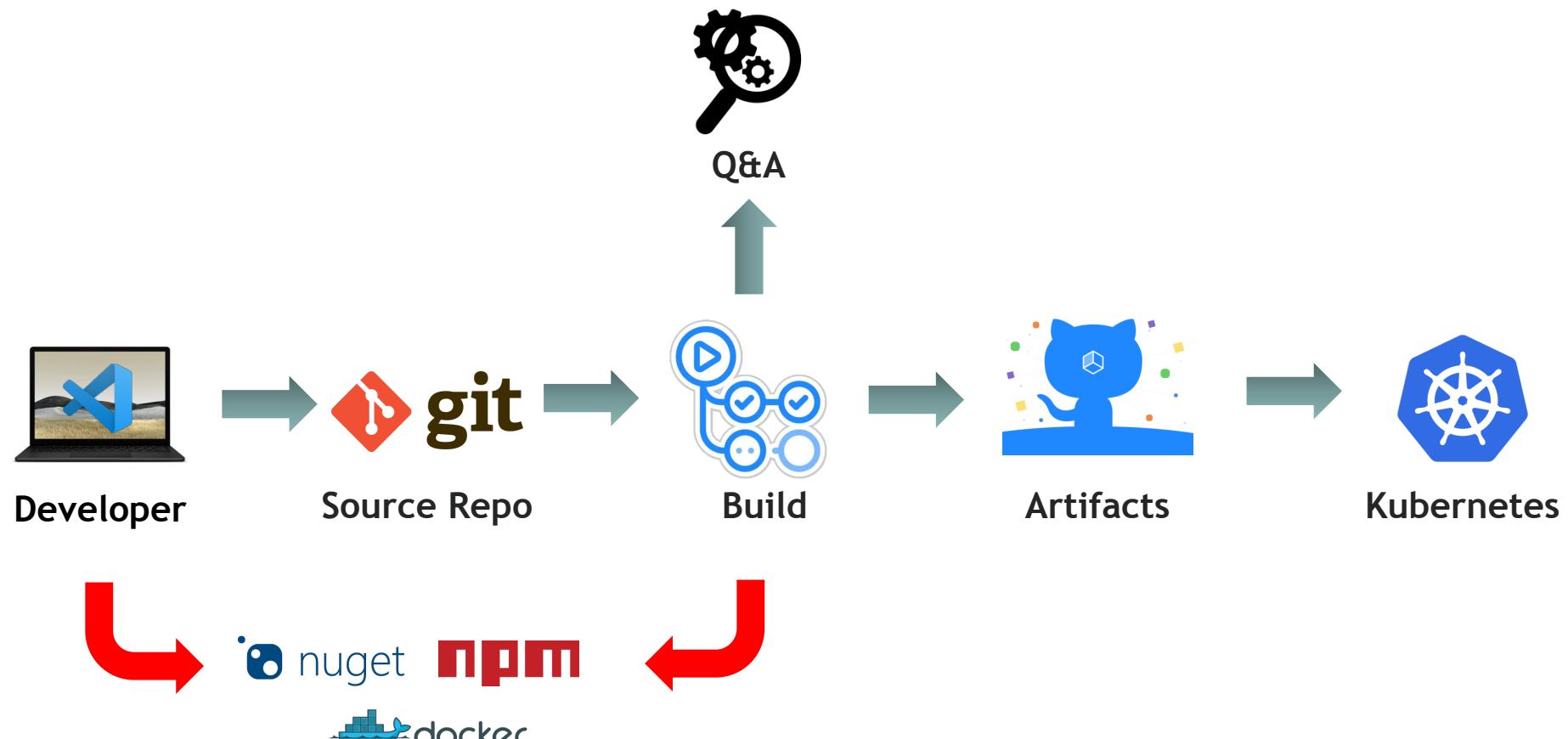
0101  
0101

# Visual Studio Code



# 3rd Party Libraries

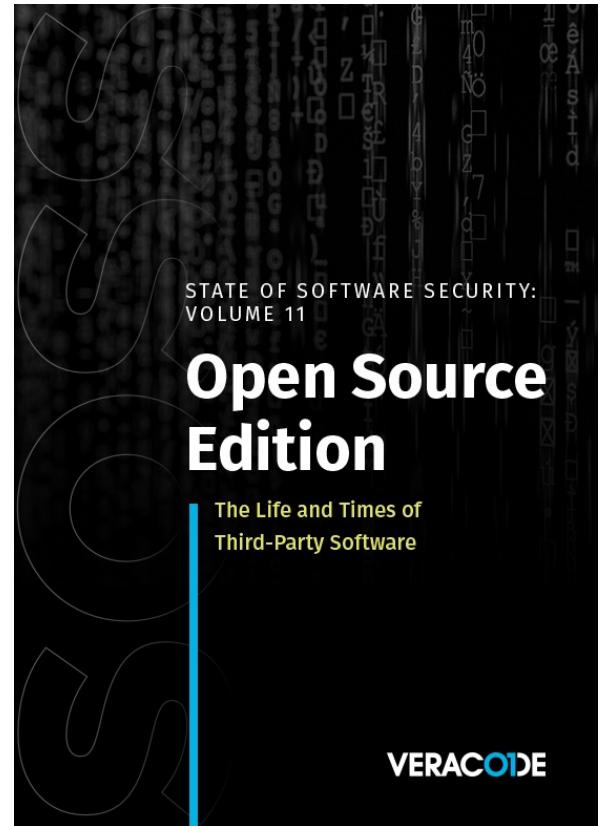
0101  
0101



# State Of Software Security v11 2021

0101  
0101

*“Despite this dynamic landscape,  
79 percent of the time, developers  
never update third-party libraries after  
including them in a codebase.”*



0101  
0101

# Vulnerabilities in libraries

A screenshot of a GitHub issue page for Microsoft Security Advisory CVE 2022-29145. The page title is "Microsoft Security Advisory CVE 2022-29145 | .NET Denial of Service Vulnerability #222". The issue was opened by dcwittaker 29 days ago with 0 comments. A comment from dcwittaker is shown, reiterating the title of the advisory. The advisory summary states: "Microsoft is releasing this security advisory to provide information about a vulnerability in .NET 6.0, .NET 5.0 and .NET Core 3.1. This advisory also provides guidance on what developers can do to update". The issue has no assignees, labels including "Monthly-Update", ".NET Core 3.1", ".NET 5.0", ".NET 6.0", and "Security", and no projects or milestones.

0101  
0101

# Vulnerabilities in libraries



Alerts and Tips    Resources    Industrial Control Systems

National Cyber Awareness System > Current Activity > Malware Discovered in Popular NPM Package, ua-parser-js

## Malware Discovered in Popular NPM Package, ua-parser-js

Original release date: October 22, 2021



Versions of a popular NPM package named [ua-parser-js](#) was found to contain malicious code. [ua-parser-js](#) is used in apps and websites to discover the type of device or browser a person is using from User-Agent data. A computer or device with the affected software installed or running could allow a remote attacker to obtain sensitive information or take control of the system.

CISA urges users and administrators using compromised ua-parser-js versions 0.7.29, 0.8.0, and 1.0.0 to update to the respective patched versions: 0.7.30, 0.8.1, 1.0.1

For more information, see [Embedded malware in ua-parser-js](#).

0101  
0101

# Vulnerabilities in libraries

The image shows two side-by-side screenshots of a Mac OS X desktop environment displaying GitHub blog posts. Both posts are from the 'Open Source' and 'Security' categories.

**Post 1: GitHub's commitment to npm ecosystem security**

**Post 2: Enrolling all npm publishers in enhanced login verification and next steps for two-factor authentication enforcement**

Both posts include a large blue cartoon illustration of a character working on a yellow puzzle piece, symbolizing security and ecosystem integration.

0101  
0101

# Dependency Confusion

A screenshot of a web browser window showing a Medium article. The title of the article is "Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies" by Alex Birsan. The article is described as "The Story of a Novel Supply Chain Attack". Below the title, there is a small profile picture of the author, Alex Birsan, and a timestamp indicating it was published on Feb 9. To the right of the author's name are social sharing icons for Twitter, Facebook, LinkedIn, and others. Below the article title is a large, colorful photograph of several shipping containers stacked together against a clear blue sky.



# Microsoft Whitepaper

## 3 ways to mitigate risk when using private package feeds

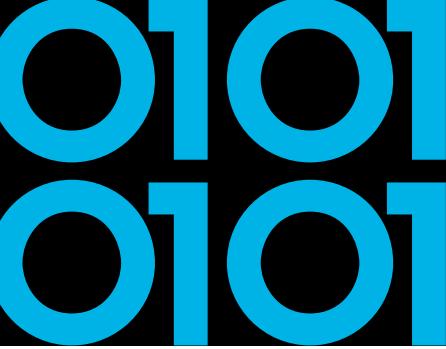
**Secure Your Hybrid Software Supply Chain**

An always-up-to-date version of this whitepaper is located at: <https://aka.ms/pkg-sec-wp>



# 3rd Party Libraries

- Intent of library, know what's inside!
- Keep in mind that's a transitive list of dependencies
- Other talk 'Sandboxing .NET Assemblies' @ NDC Porto
  
- Open Source Security Foundation - OpenSSF
- Security Scorecards - Security health metrics for Open Source



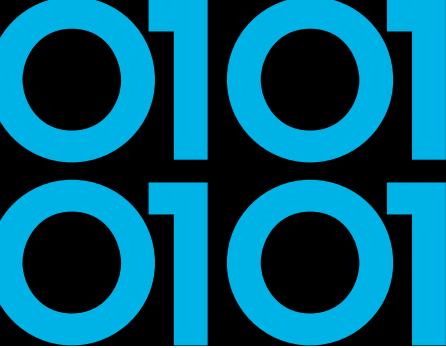
# Security Scorecards - OpenSSF

The screenshot shows a GitHub repository page for `ossf/scorecard`. The title is "ossf/scorecard: Security Score". The URL in the address bar is <https://github.com/ossf/scorecard>. The page content includes:

- A header section titled "Security Scorecards".
- Build status indicators: "build passing" and "CodeQL passing".
- An "Overview" section with two bullet points:
  - What Is Scorecards?
  - Prominent Scorecards Users
- An "Using Scorecards" section with eight bullet points:
  - Prerequisites
  - Installation
  - Authentication
  - Basic Usage
  - Report Problems
  - Scorecards' Public Data
- A "Checks" section.

On the right side of the page, there is a large cartoon illustration of a blue bird-like character wearing red shorts and a yellow shield, holding up a yellow card with the number "10" on it.

# Deps.Dev by Google



The screenshot shows a dark-themed web browser window for 'Open Source Insights' at <https://deps.dev>. At the top, a green banner reads 'Open Source Insights is an experimental project by Google.' Below it, the main header features the text 'open/source/insights' in white. To the right are links for 'About', 'Documentation', and 'Blog'. The central content area has a dark grey background with white text. A large heading 'Understand your dependencies' is followed by a paragraph explaining the importance of dependency management. On the right side, there's a list of dependency statistics with icons: npm packages (1.80M), Go modules (717k), Maven artifacts (427k), PyPI packages (325k), Cargo crates (72k), and NuGet packages (Coming soon). At the bottom left is a search bar with a magnifying glass icon and the placeholder 'Search for open source packages'. Next to it are dropdown menus for 'All systems' and a blue 'Search' button.

/ npm PACKAGES	1.80M
/ Go MODULES	717k
/ Maven ARTIFACTS	427k
/ PyPI PACKAGES	325k
/ Cargo CRATES	72k
/ NuGet PACKAGES	Coming soon

# Deps.Dev by Google



[electron/electron](#)

GitHub

:electron: Build cross-platform desktop apps with  
JavaScript, HTML, and CSS

14k forks

102k stars

## OpenSSF scorecard

The [Open Source Security Foundation](#) is a cross-industry collaboration to improve the security of open source software (OSS). The [Scorecard](#) provides security health metrics for open source projects.

[View information about checks and how to fix failures.](#)

SCORE

5.8/10

Scorecard as of April 25, 2022.

▶ Code-Review	5/10
▶ Maintained	10/10
▶ CII-Best-Practices	0/10
▶ Vulnerabilities	10/10
▶ Dependency-Update-Tool	0/10
▶ Security-Policy	10/10
▶ Dangerous-Workflow	10/10
▶ Token-Permissions	0/10
▶ License	10/10
▶ Pinned-Dependencies	8/10
▶ Binary-Artifacts	10/10
▶ Fuzzing	0/10
▶ Signed-Releases	0/10

Project metadata as of May 7, 2022.



# Reproducible/Deterministic Builds

The screenshot shows a website with a dark blue header featuring the Reproducible Builds logo and a navigation menu on the left. The menu includes links for Home, Contribute, Documentation (which is currently selected), Tools, Who is involved?, News, Events, and Talks. The main content area contains a large heading 'Definitions' and a sub-section titled 'When is a build reproducible?' with its definition.

## Definitions

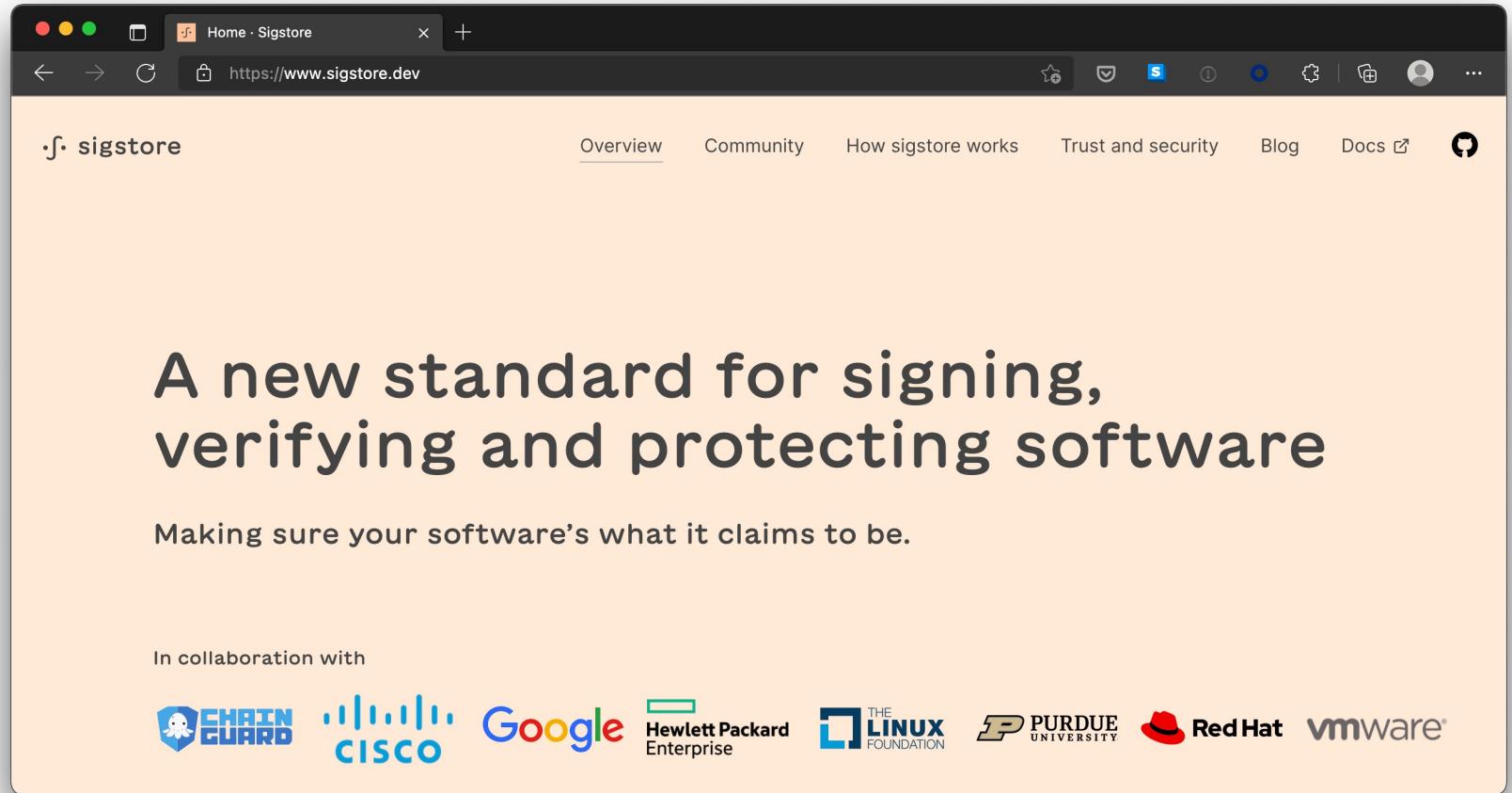
### When is a build reproducible?

A build is **reproducible** if given the same source code, build environment and build instructions, any party can recreate bit-by-bit identical copies of all specified artifacts.

The relevant attributes of the build environment, the build instructions and the source code as well as the expected reproducible artifacts are defined by the authors or distributors. The artifacts of a build are the parts of the build results that are the desired primary output.

# Signing artifacts

0101  
0101



A screenshot of a web browser displaying the Sigstore website at <https://www.sigstore.dev>. The page has a light orange background. At the top, there's a navigation bar with links for Overview, Community, How sigstore works, Trust and security, Blog, and Docs. Below the navigation, the main heading reads "A new standard for signing, verifying and protecting software" in large, bold, dark gray font. Underneath it, a subtitle says "Making sure your software's what it claims to be." At the bottom, there's a section titled "In collaboration with" featuring logos for ChainGuard, Cisco, Google, Hewlett Packard Enterprise, The Linux Foundation, Purdue University, Red Hat, and VMware.

0101  
0101

# Signing artifacts

The screenshot shows a web browser window for the 'Sigstore' website at <https://www.sigstore.dev>. The page title is 'How sigstore works'. The content includes a section about the benefits of using sigstore, a 'Standardized approach' section, and a 'Building for future integrations' section. To the right of the content is a diagram illustrating the sigstore architecture. The diagram shows a flow from 'DEVELOPERS, MAINTAINERS, MONITORS' through three main steps: 'SIGN AND PUBLISH ARTIFACTS', 'PUBLISH SIGNING CERTIFICATES', and 'MONITOR LOGS'. These steps interact with a 'FULCIO CERTIFICATE AUTHORITY', a 'SIGNATURE TRANSPARENCY LOG', and a 'KEY TRANSPARENCY LOG'. All components are connected to a central 'TRUST ROOT'.

How sigstore works

sigstore is a set of tools developers, software maintainers, package managers and security experts can benefit from. Bringing together free-to-use open source technologies like Fulcio, Cosign and Rekor, it handles digital signing, verification and checks for provenance needed to make it safer to distribute and use open source software.

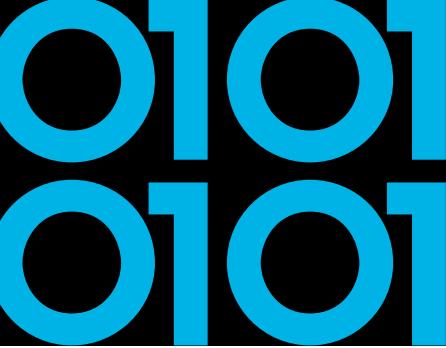
A standardized approach

This means that open source software uploaded for distribution has a stricter, more standardized way of checking who's been involved, that it hasn't been tampered with. There's no risk of key compromise, so third parties can't hijack a release and slip in something malicious.

Building for future integrations

With the help of a working partnership that includes Google, the Linux Foundation, Red Hat and Purdue University, we're in constant collaboration to find new ways to improve the sigstore technology, to make it easy to adopt, integrate and become a long-lasting standard.

```
graph TD; Developers[DEVELOPERS, MAINTAINERS, MONITORS] --> Sign[SIGN AND PUBLISH ARTIFACTS]; Developers --> Publish[PUBLISH SIGNING CERTIFICATES]; Developers --> Monitor[MONITOR LOGS]; Sign -.-> Fulcio[FULCIO CERTIFICATE AUTHORITY]; Publish -.-> Signature[SIGNATURE TRANSPARENCY LOG]; Monitor -.-> Key[KEY TRANSPARENCY LOG]; Fulcio --- TRUST_ROOT[TRUST ROOT]; Signature --- TRUST_ROOT; Key --- TRUST_ROOT;
```

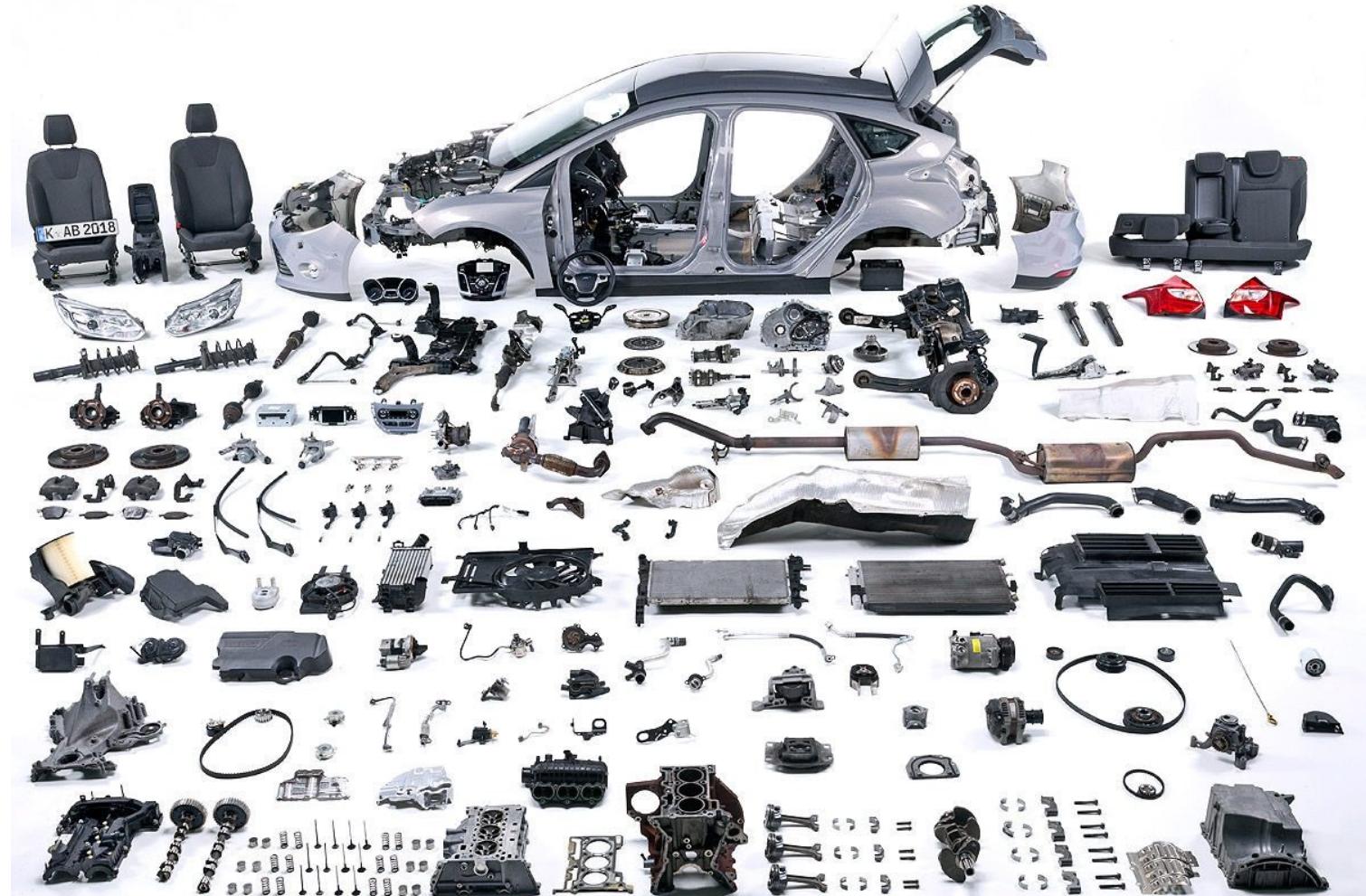


# Signing artifacts

- Cosign can be used for signing Docker images
- It can do keyless signing based on OpenID Connect
- GitHub Actions have released OpenID Connect support since end 2021

0101  
0101

# Automotive Industry



# Car Supply Chain

0101  
0101



## Tata Steel Factory

- Iron Ore from Sweden
- ISO 6892-1 Tested/Certified
  - Batch #1234

## Bosch Factory

- Steel Batch #1234 Tata
- ECE-R90 Tested/Certified
  - Serie #45678
- Used by Ford, Volkswagen and Renault

## Renault Manufacturing

- Bosch Disk #45678
- Bosal Exhaust #RE9876
- Goodyear Tires #GY8877
- Kadjar VIN 1234567890

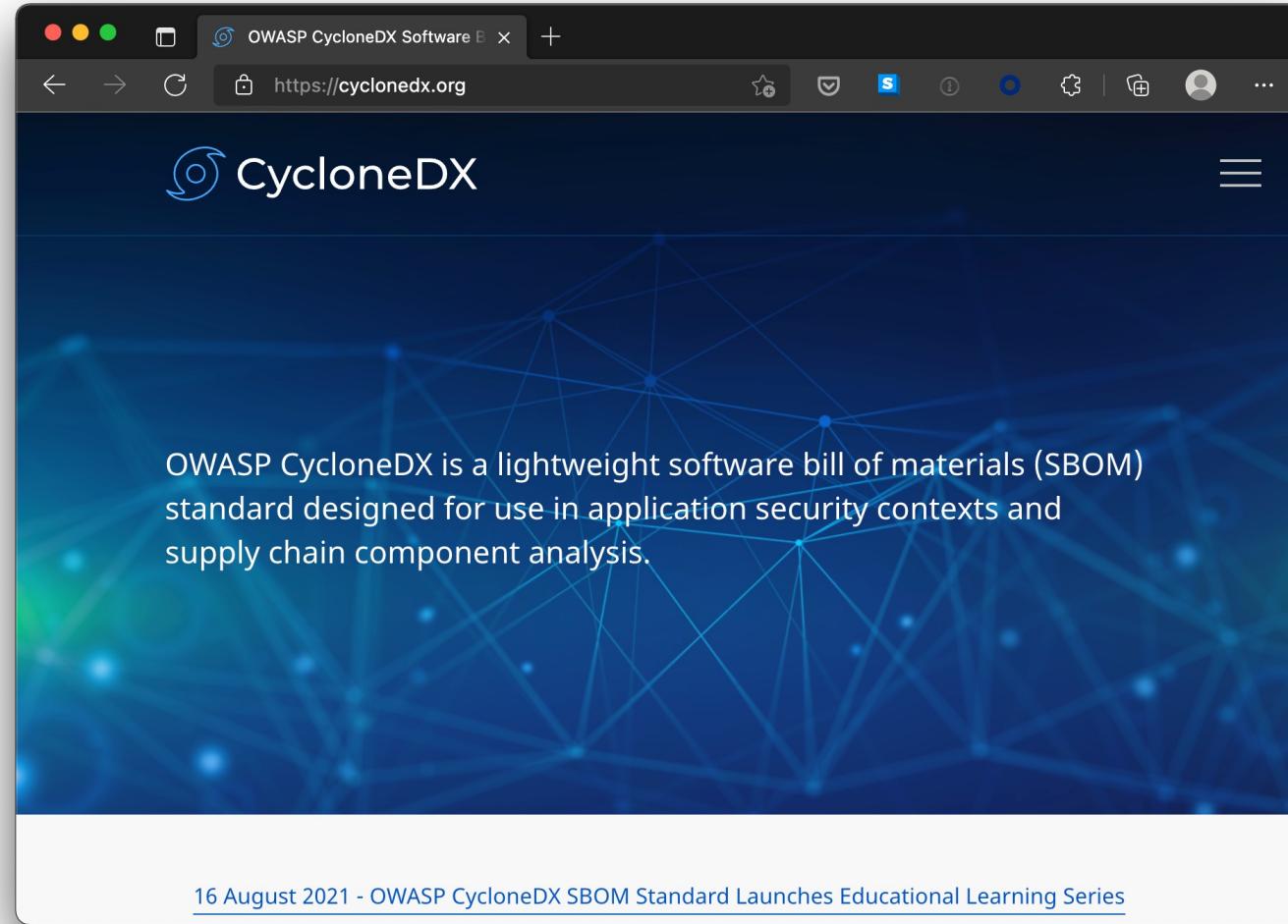


# Software Bill of Materials (SBOM)

- Industry standard of describing the software
  - Producer Identity - Who Created it?
  - Product Identity - What's the product?
  - Integrity - Is the project unaltered?
  - Licensing - How can the project be used?
  - Creation - How was the product created?
  - Materials - How was the product created?

0101  
0101

# Software Bill of Materials (SBOM)



0101  
0101

# Docker SBOM

The screenshot shows a web browser displaying a Docker blog post. The title of the post is "Announcing Docker SBOM: A step towards more visibility into Docker images". The author is listed as JUSTIN CORMACK, dated Apr 7 2022. The post content discusses Docker's first step in making container images more visible, specifically mentioning the experimental `docker sbom` CLI command. To the right of the post, there are sections for "Post Tags" and "Categories", each with a list of related topics.

Join us for [DockerCon](#) on May 9-10th. Preview the agenda and [register today](#).

Products Developers Pricing Blog About Us Partners

Get Started

## Announcing Docker SBOM: A step towards more visibility into Docker images

JUSTIN CORMACK Apr 7 2022

Today, Docker takes its first step in making what is inside your container images more visible so that you can better secure your software supply chain. Included in Docker Desktop 4.7.0 is a new, experimental `docker sbom` CLI command that displays the SBOM (Software Bill Of Materials) of any Docker image. It will

Post Tags

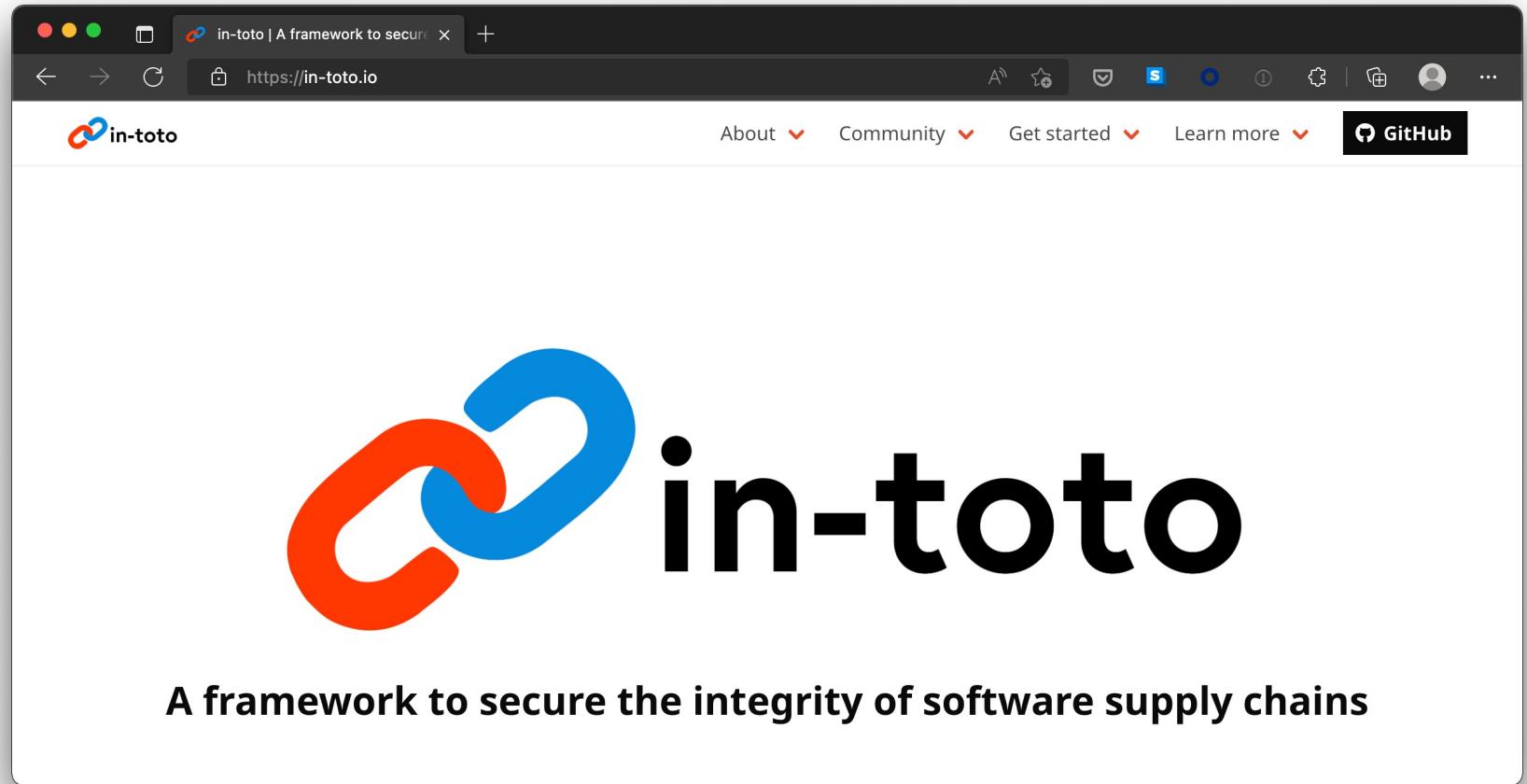
- # docker
- # Docker images
- # sbom
- # Secure Software Supply Chain

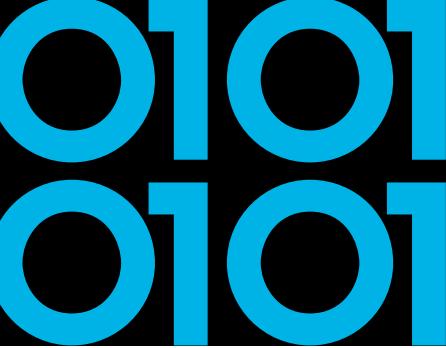
Categories

- Community
- Company
- Engineering
- Newsletters
- Products

# In-toto

0101  
0101





# In-Toto - Demo

## In-Toto - Demo - Terminology



Niels Tanis

- **Functionaries** that are identified by public key our supply chain.  
Niels (Project-Owner), Aimee (Developer) and Noud (Packager)
- **Project-Owner** defines a (**Supply Chain**) **Layout** that describes **what** happens and by **who** and what the produced **Materials** and **Byproducts** are
- Link metadata is output of executed step in the **Layout**  
**Materials** are input, **Products** are output and can be used as **Materials** in later steps

+17

0101  
0101

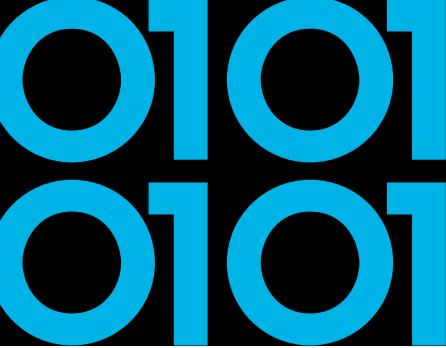
# MyAwesomeWebApp Demo

- GitHub Actions
- In-toto
- Sigstore Cosign
- Docker SBOM with Syft

0101  
0101

# Google SLSA

The screenshot shows a web browser window displaying the SLSA (Supply-chain Levels for Software Artifacts) website at <https://slsa.dev>. The page has a dark header with a light background. The main heading reads "Improving artifact integrity across the supply chain". Below it, a definition of SLSA is provided: "SLSA ("salsa") is Supply-chain Levels for Software Artifacts. A security framework from source to service, giving anyone working with software a common language for increasing levels of software security and supply chain integrity." At the bottom of the page, there is a navigation bar with links to "Overview", "Security levels", "Requirements", "Use cases", "Provenance", "Roadmap", and "Get involved".



# Google SLSA Levels

Level	Description	Example
1	Documentation of the build process	Unsigned provenance
2	Tamper resistance of the build service	Hosted source/build, signed provenance
3	Extra resistance to specific threats	Security controls on host, non-falsifiable provenance
4	Highest levels of confidence and trust	Two-party review + hermetic builds

0101  
0101

# SUSE SLSA Level 4

The screenshot shows a web browser window with the following details:

- Title Bar:** The title bar displays "SLSA: Securing the Software Supply Chain" and the URL "https://documentation.suse.com/sbp/server-linu...".
- Header:** The header includes the SUSE logo and a breadcrumb navigation: "SLSA: Securing the Software Supply Chain".
- Section Header:** A large green section header reads "All SUSE Products" followed by "SLSA: Securing the Software Supply Chain".
- Abstract:** A green "Abstract" button is present.
- Text Content:** The main content area contains a paragraph about SUSE's role in software supply chain security and its preparation for SLSA L4 compliance. It also includes a "REPORT DOCUMENTATION BUG" link.
- Disclaimer:** A detailed disclaimer at the bottom states that the document is part of the SUSE Best Practices series, contributed voluntarily by SUSE employees and third parties, and that SUSE cannot verify the accuracy or consequences of the actions described.

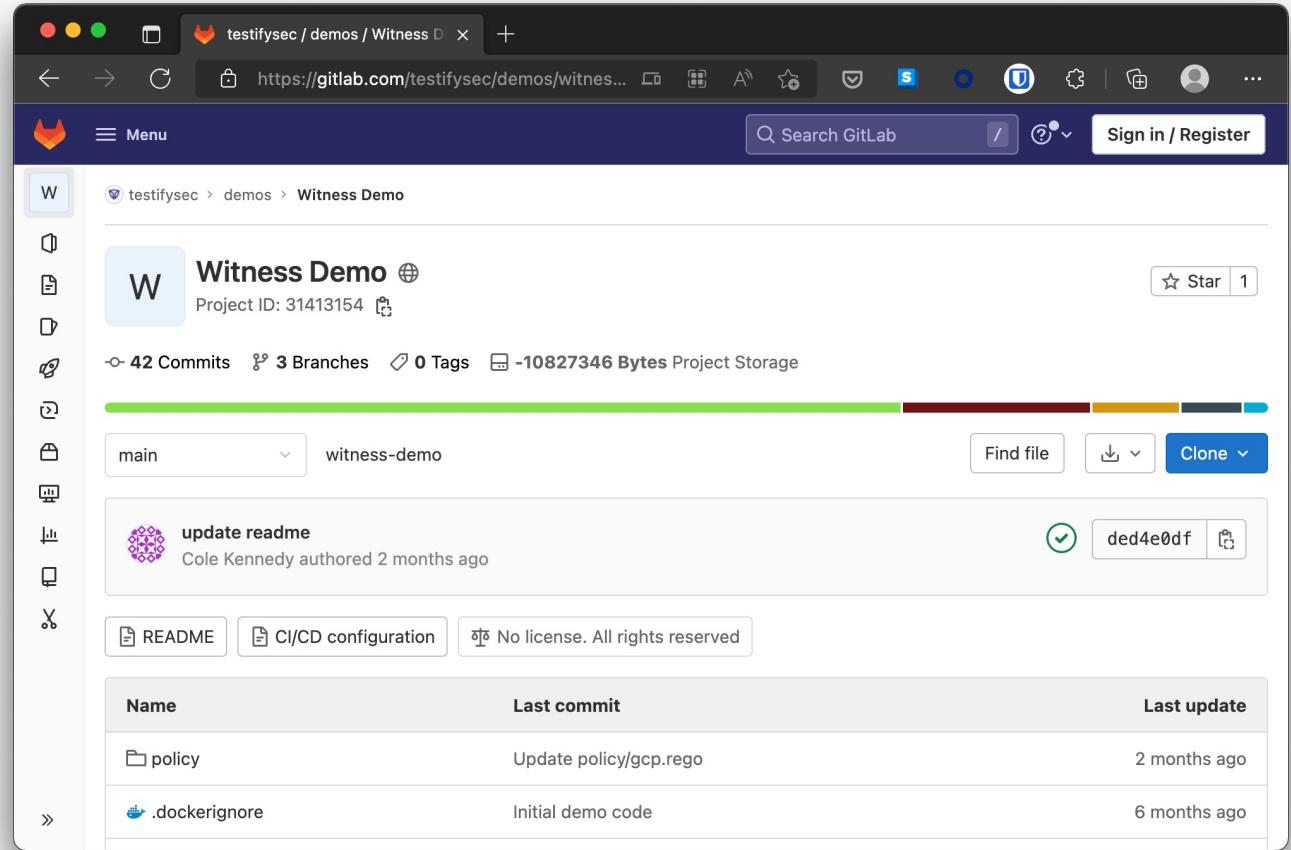


# SLSA GitHub Action

- Released April 2022
- SLSA level 2 provenance generator in GitHub Action
- SLSA level 3+ provenance generator for Go binaries
  - GitHub Hosted Runner
  - Uses SigStore to do keyless signing with GitHub OIDC
  - Verifier included

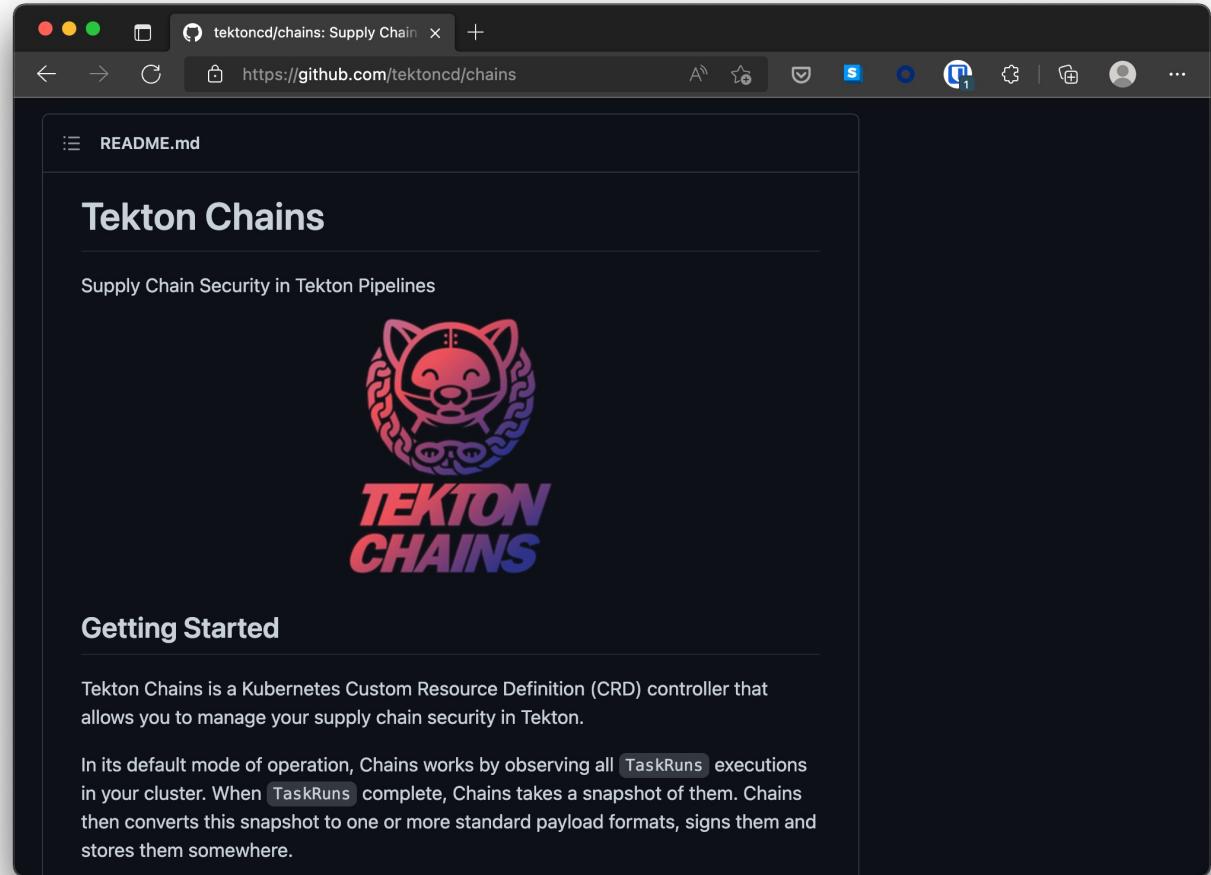
0101  
0101

# Witness & GitLab Attestator



0101  
0101

# Open Shift - Tekton - Tekton Chains

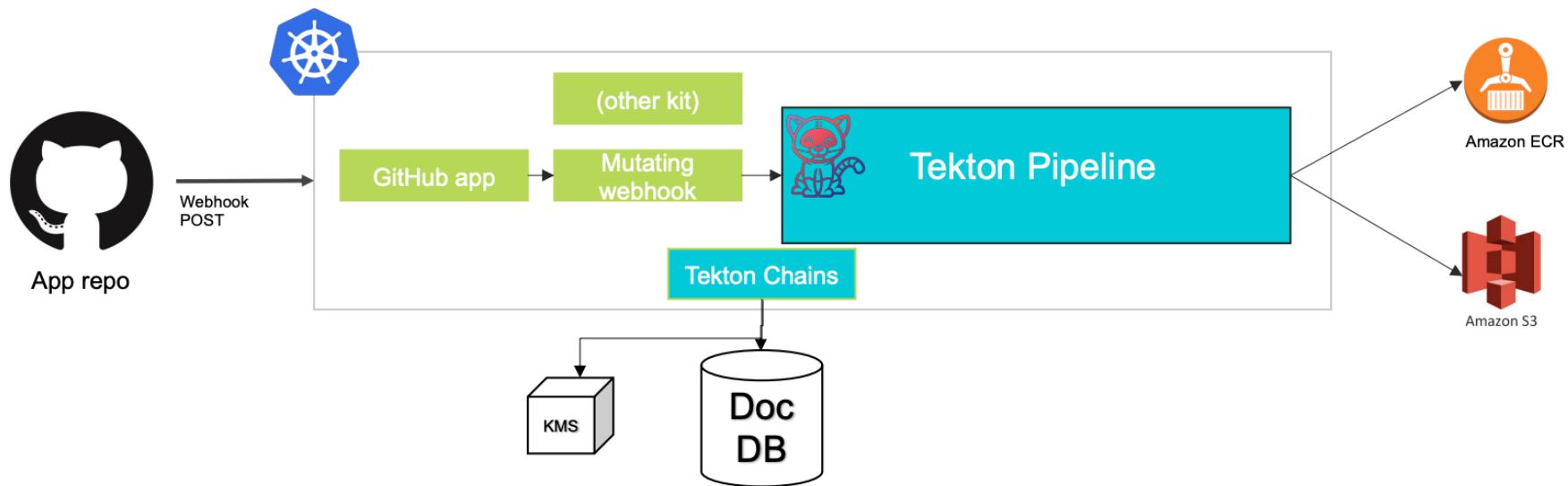




# SolarWinds Project Trebuchet



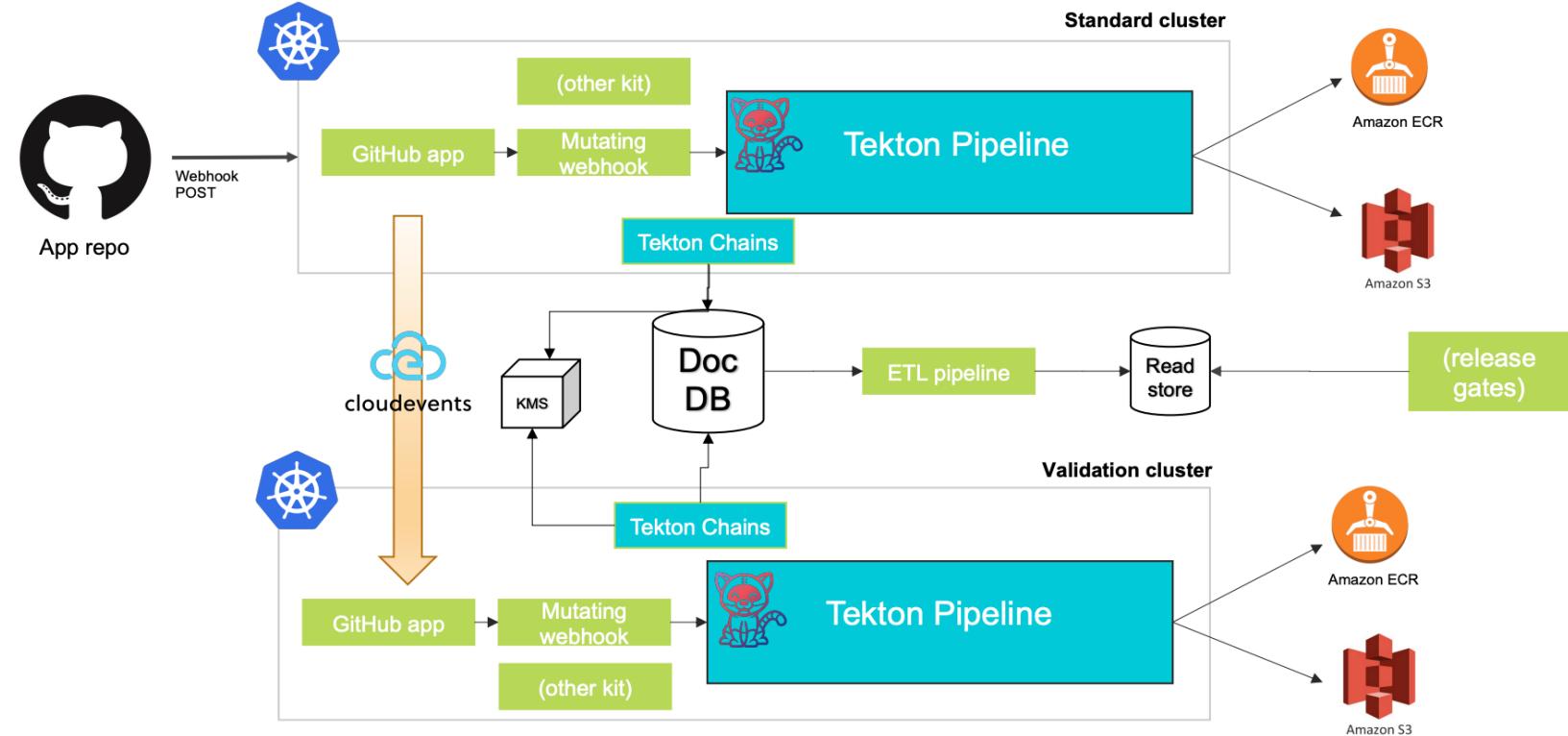
## Pipeline With Attestations



# SolarWinds Project Trebuchet



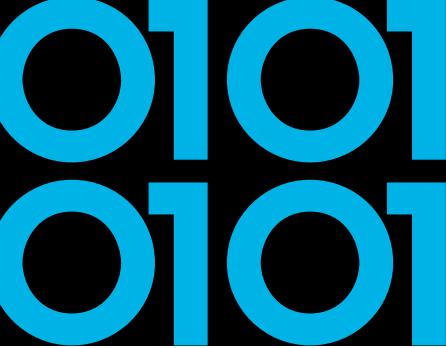
## Reading Results





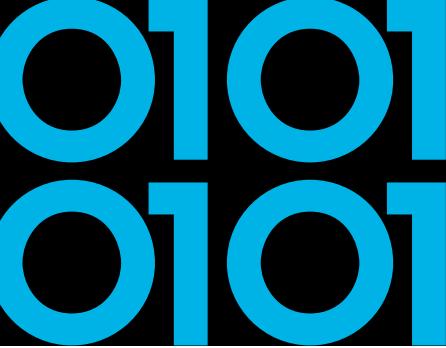
# Working with SBOM

- Kyverno Policy Management
- Chainguard Enforce
- Google Grafeas & Kritis
- Azure Policy?



# Conclusion

- It's not how it's more a matter of when!
- Be aware of your used software supply chain(s).
- Know what you're using and pulling into projects.



# Conclusion

- Integrate security into your software lifecycle.
- Start working on creating SBOM's and see how SLSA can fit into your process.
- Try to work with SBOM output and use it!

VERACODE

Thanks! Questions?

<https://github.com/nielstanis/wearedevs2022>

ntanis at veracode.com

@nielstanis on Twitter

W>

