Securing your application software supply-chain

Niels Tanis

Who am I?

- Niels Tanis
- Senior Principal Security Researcher @ Veracode
  - Background .NET Development, Pentesting/ethical hacking, and software security consultancy
  - ISC² CSSLP
  - Research on static analysis for .NET apps

WeAreDevelopers

@nielstanis

Securing your application software supply-chain

WeAreDevelopers

@nielstanis

Agenda

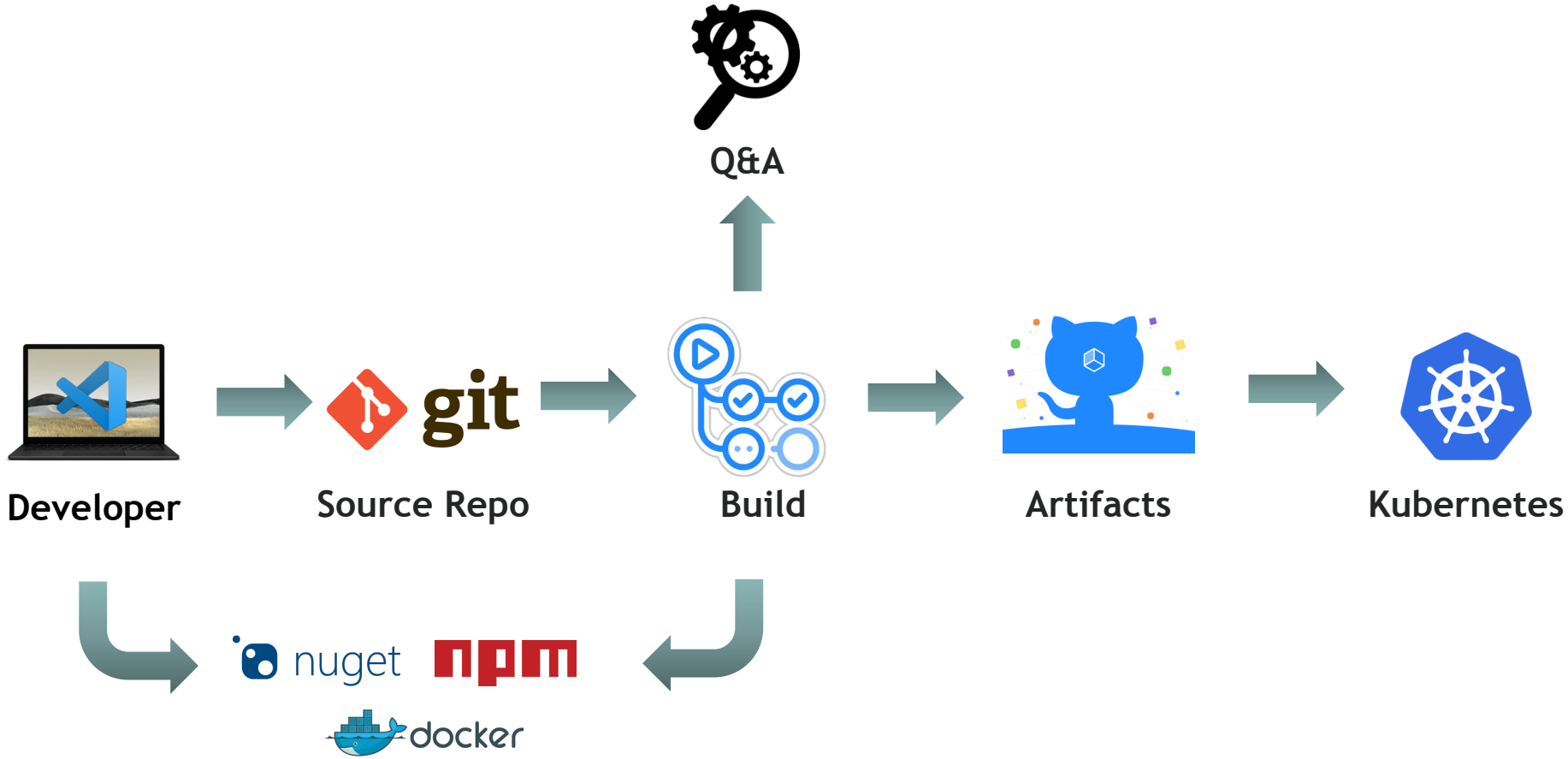
- Definition Software Supply-Chain
- Securing the Software Supply-Chain
  - Developer & Source
  - 3rd Party Libraries
  - Build & Release
- Conclusion and Q&A

What is a Supply Chain?

WeAreDevelopers

@nielstanis

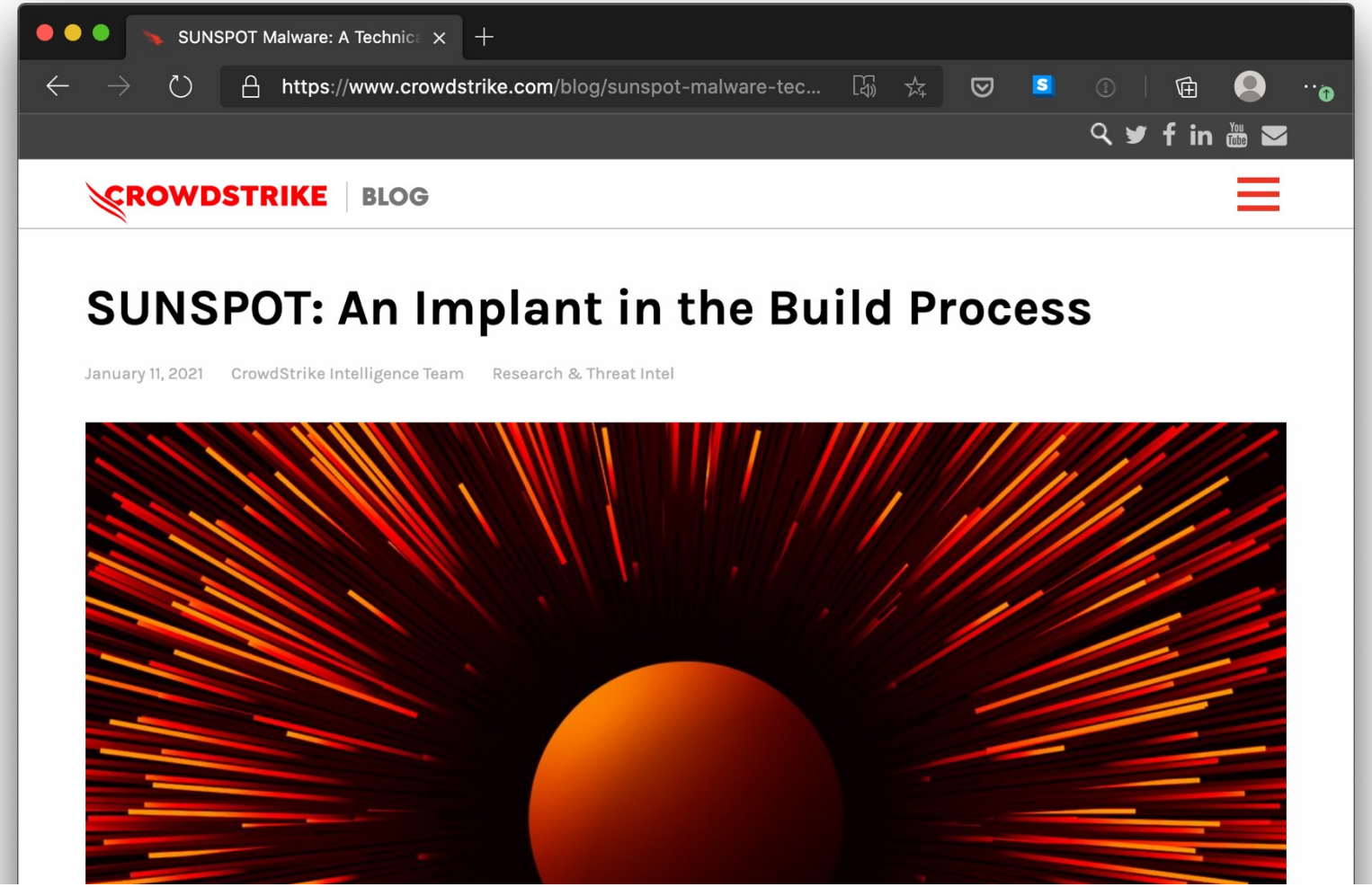
Software Supply Chain



WeAreDevelopers

@nielstanis

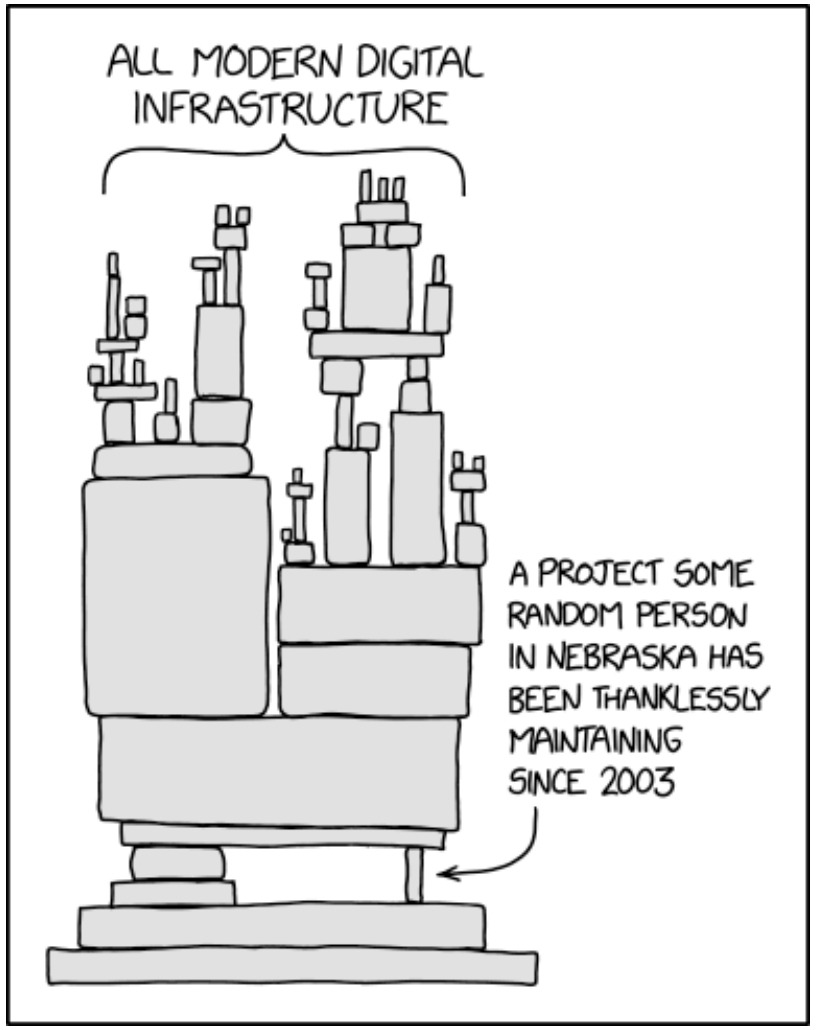
SolarWinds SunSpot



WeAreDevelopers

@nielstanis

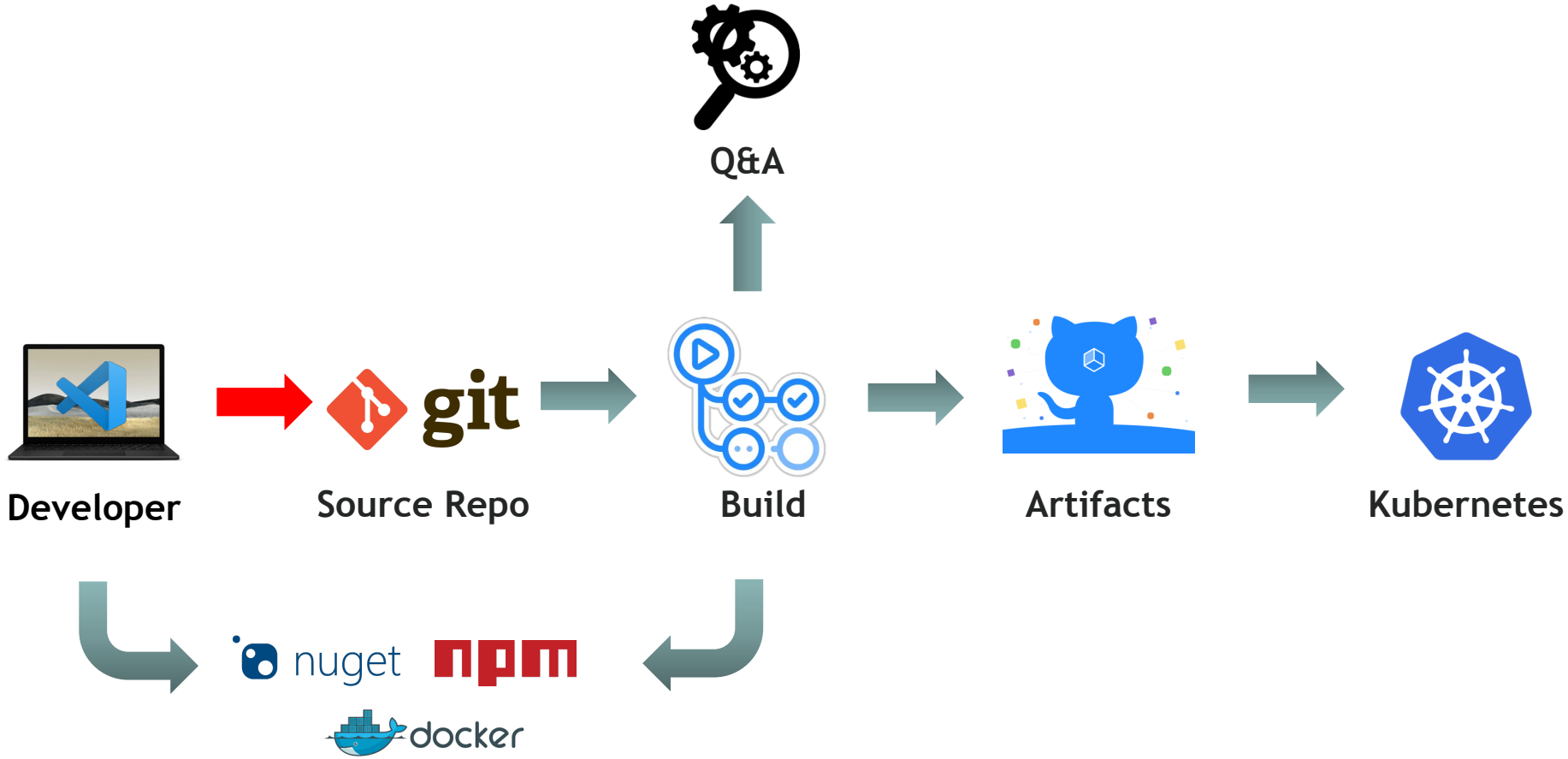
XKDC 2347 - Dependency



WeAreDevelopers

@nielstanis

Software Supply Chain



WeAreDevelopers

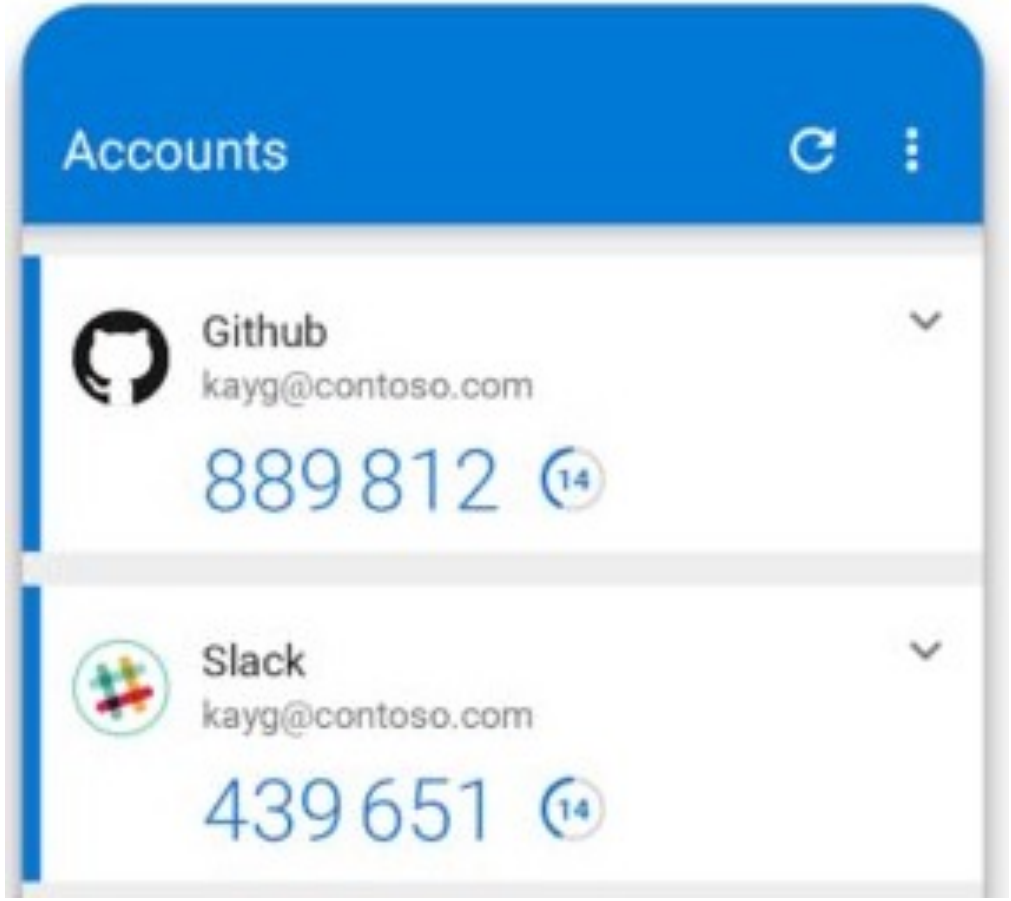
@nielstanis

GitHub account

WeAreDevelopers

@nielstanis

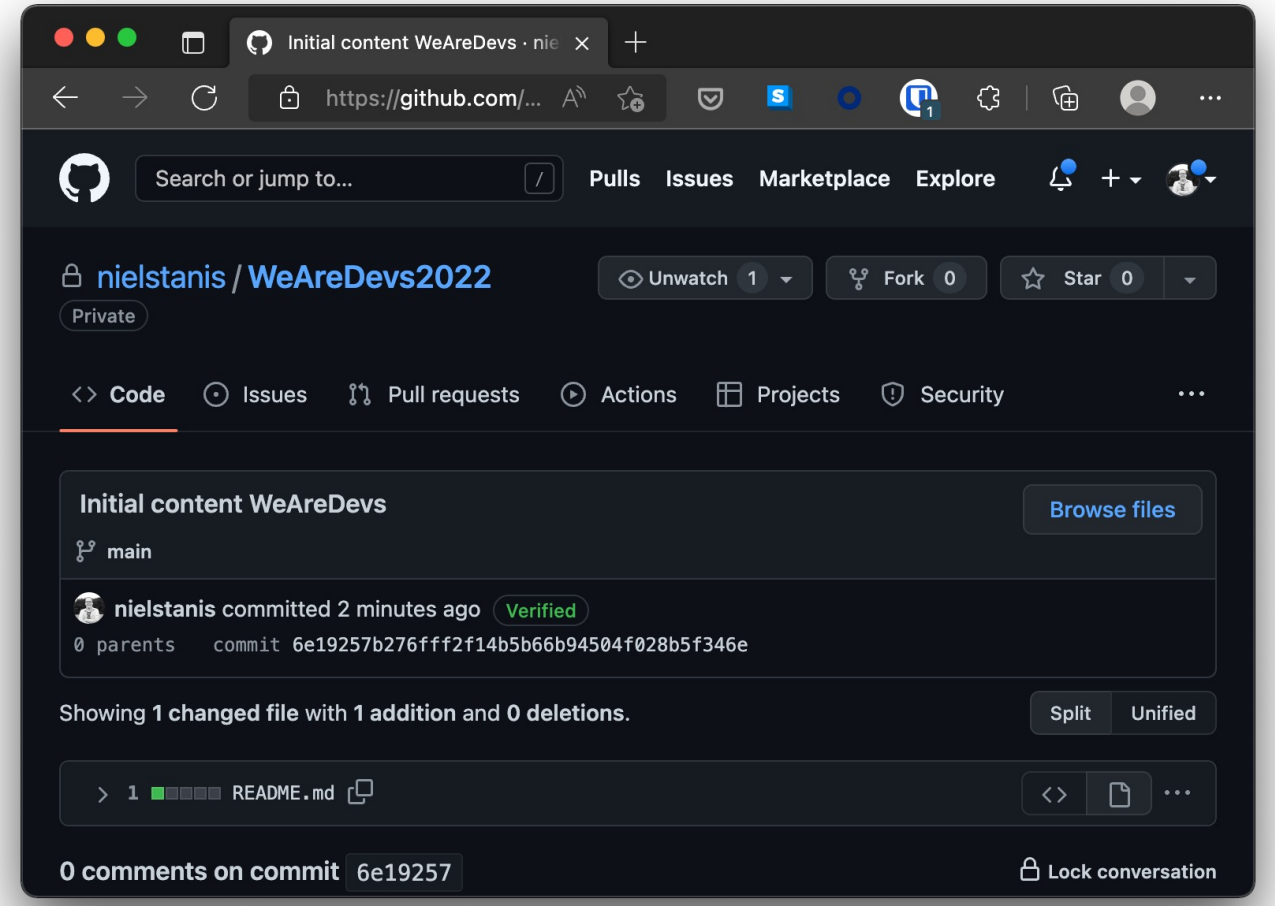
Use MFA on source-repository



WeAreDevelopers

@nielstanis

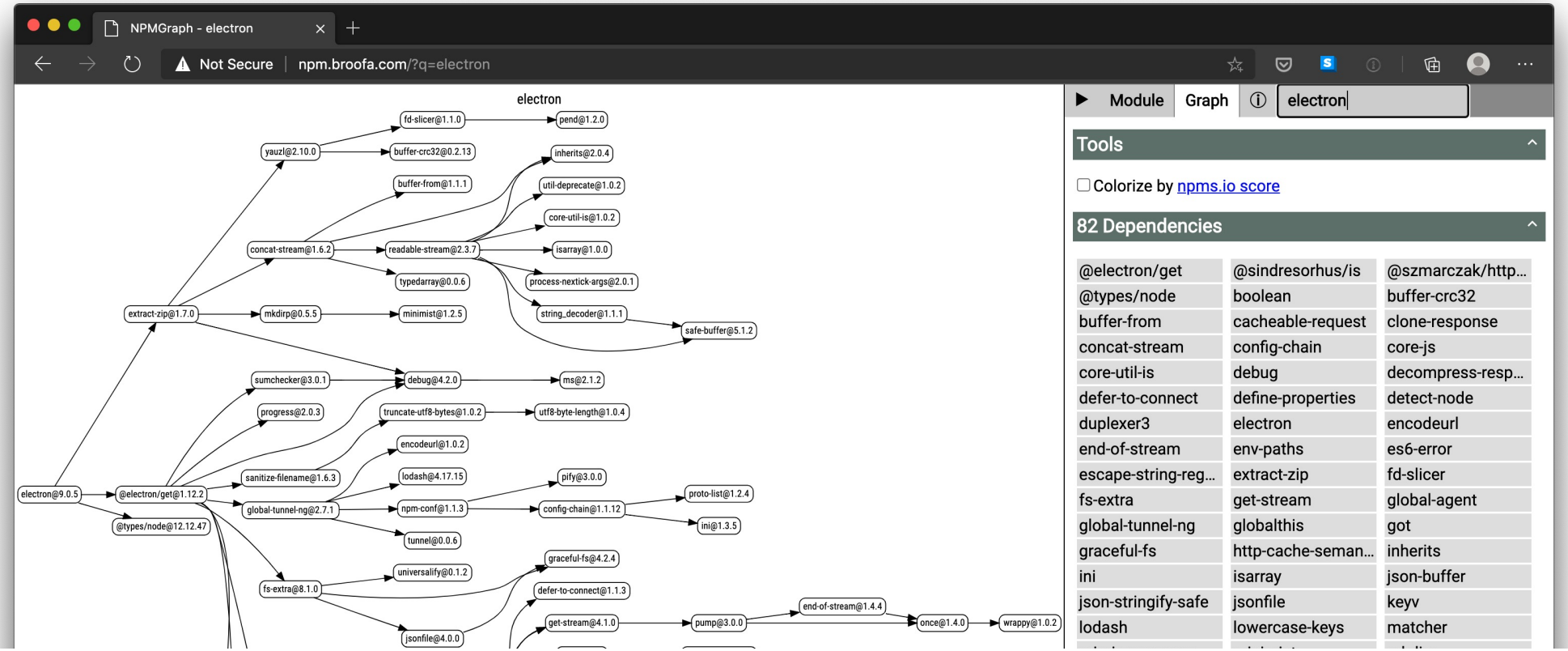
GIT Commit Signing



WeAreDevelopers

@nielstanis

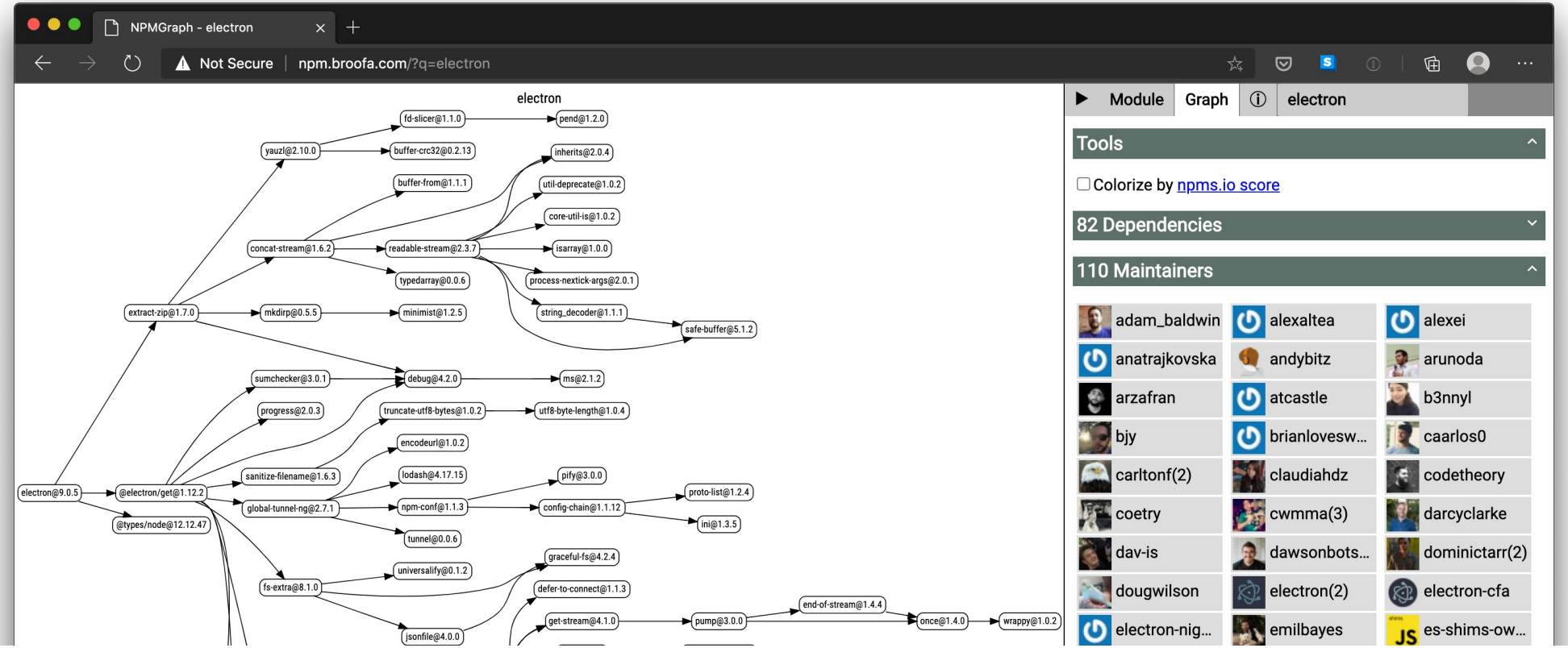
Visual Studio Code



WeAreDevelopers

@nielstanis

Visual Studio Code



WeAreDevelopers

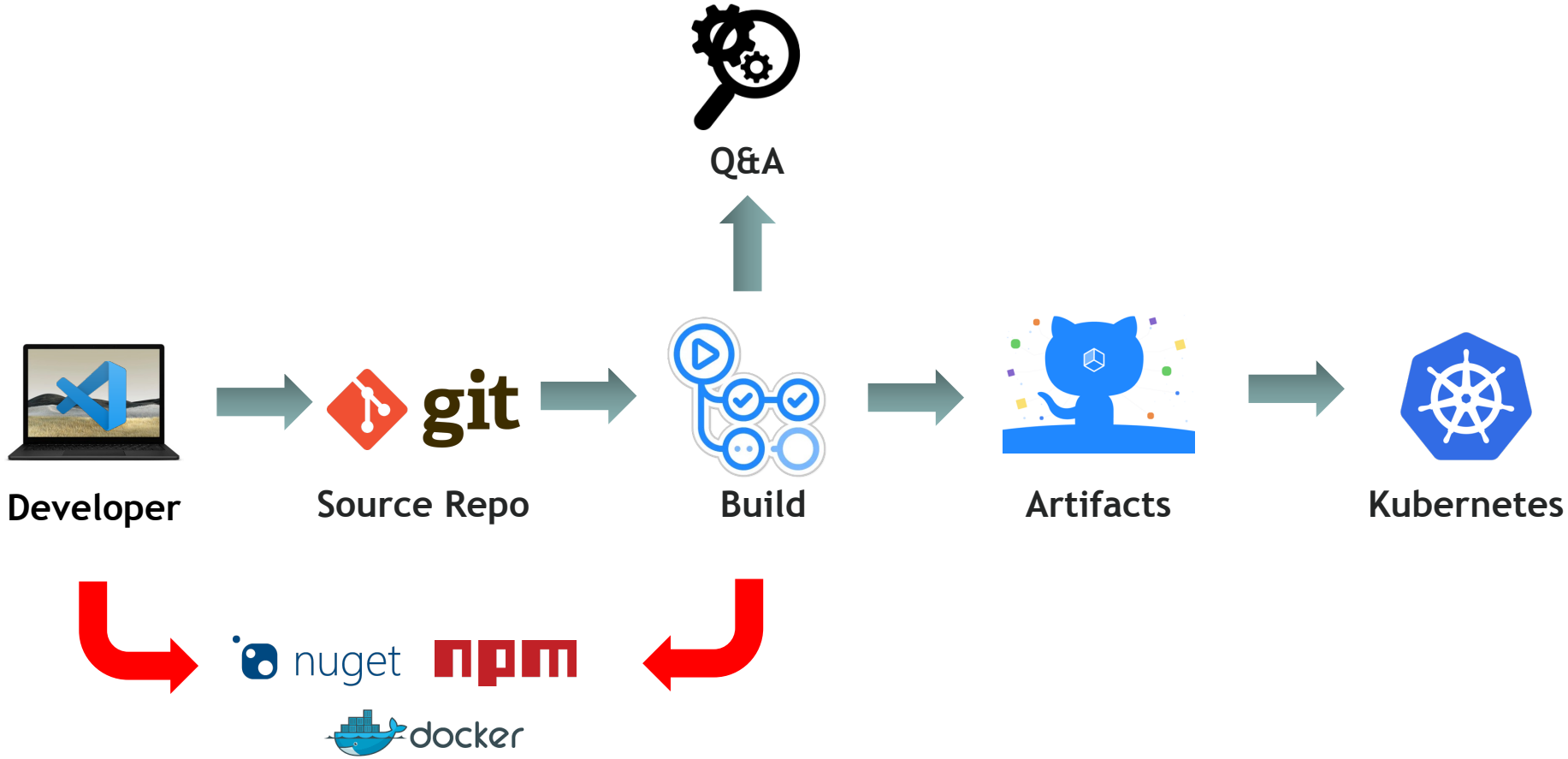
@nielstanis

Visual Studio Code

WeAreDevelopers

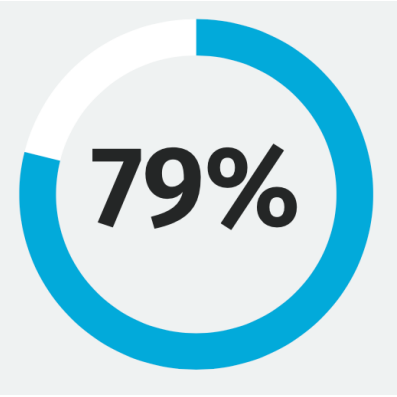
@nielstanis

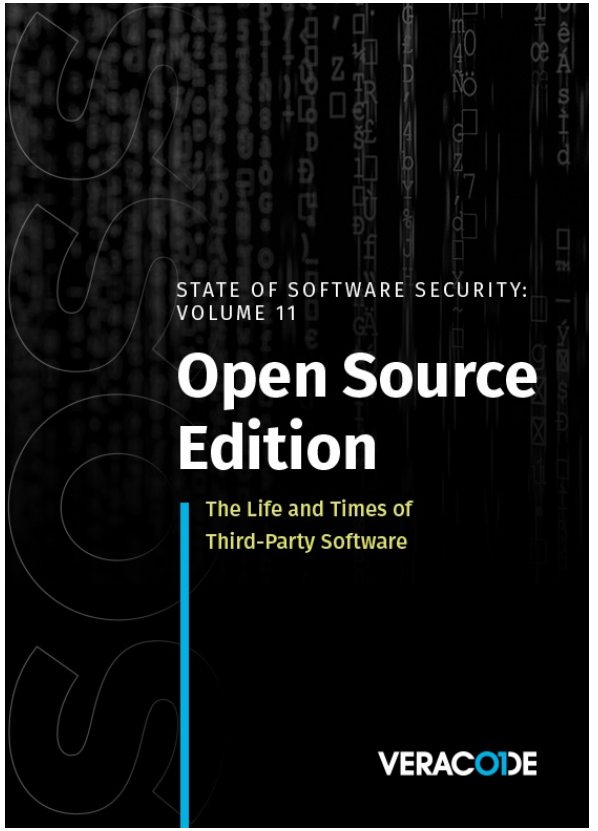
3rd Party Libraries



State Of Software Security v11 2021

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

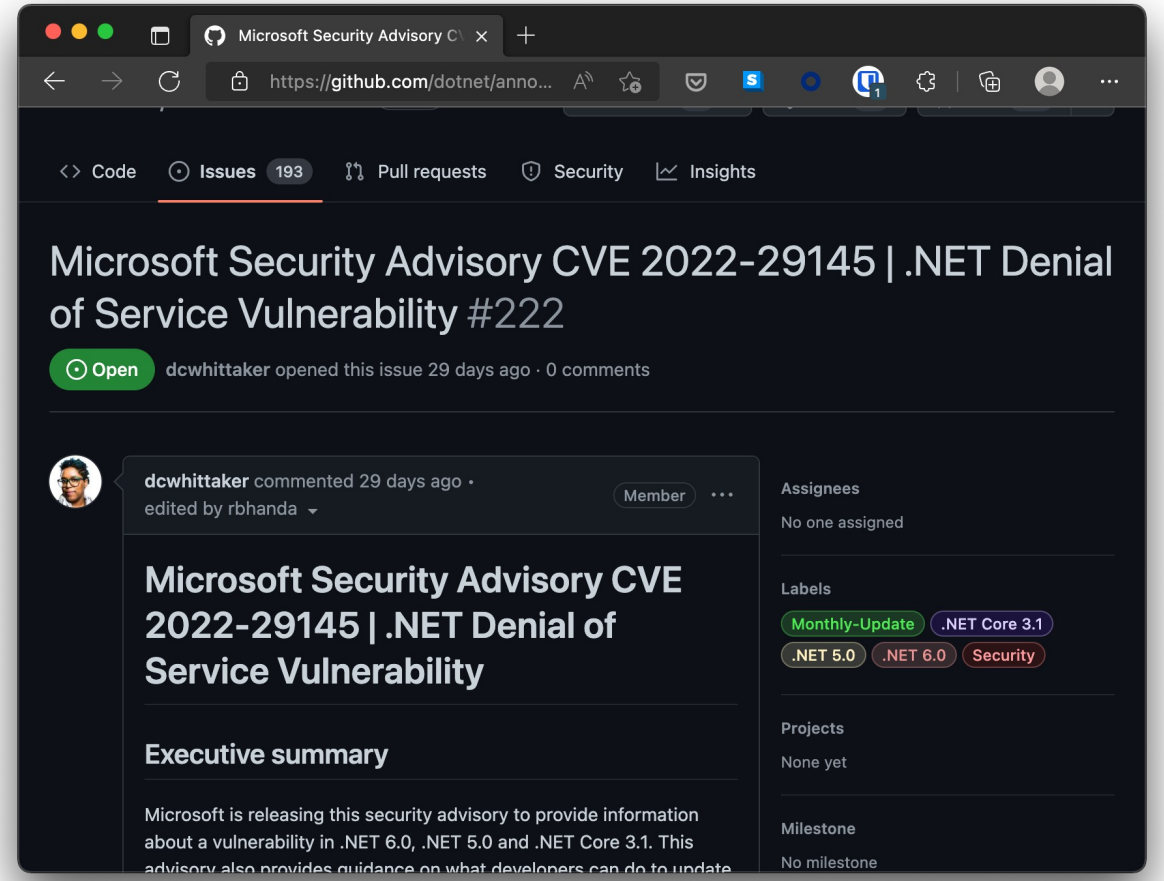




WeAreDevelopers

@nielstanis

Vulnerabilities in libraries



WeAreDevelopers

@nielstanis

Vulnerabilities in libraries

CYBERSECURITY & INFRASTRUCTURE SECURITY AGENCY

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

Print | Tweet | Send | Share

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.

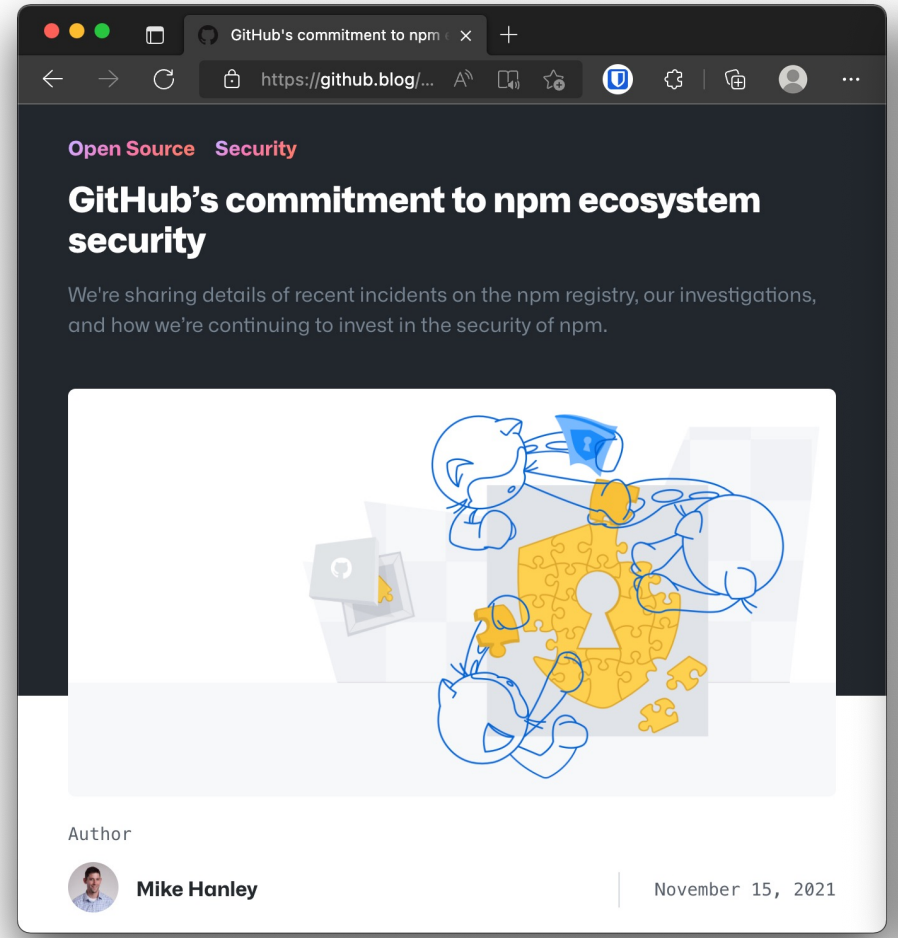
CISA urges users and administers using compromised ua-parser-js versions 0.7.29, 0.8.0, and 1.0.0 to update to the respective patched versions: 0.7.30, 0.8.1, 1.0.1

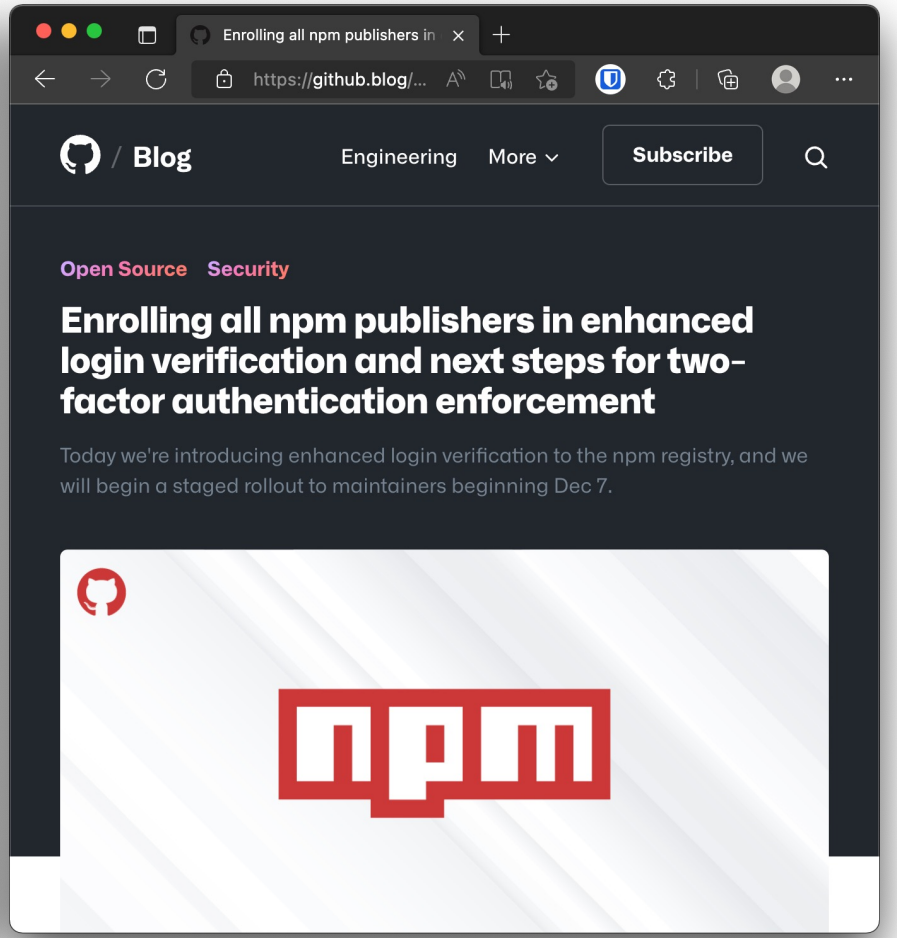
For more information, see Embedded malware in ua-parser-js.

WeAreDevelopers

@nielstanis

Vulnerabilities in libraries





WeAreDevelopers

@nielstanis

Dependency Confusion

WeAreDevelopers

@nielstanis

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

WeAreDevelopers

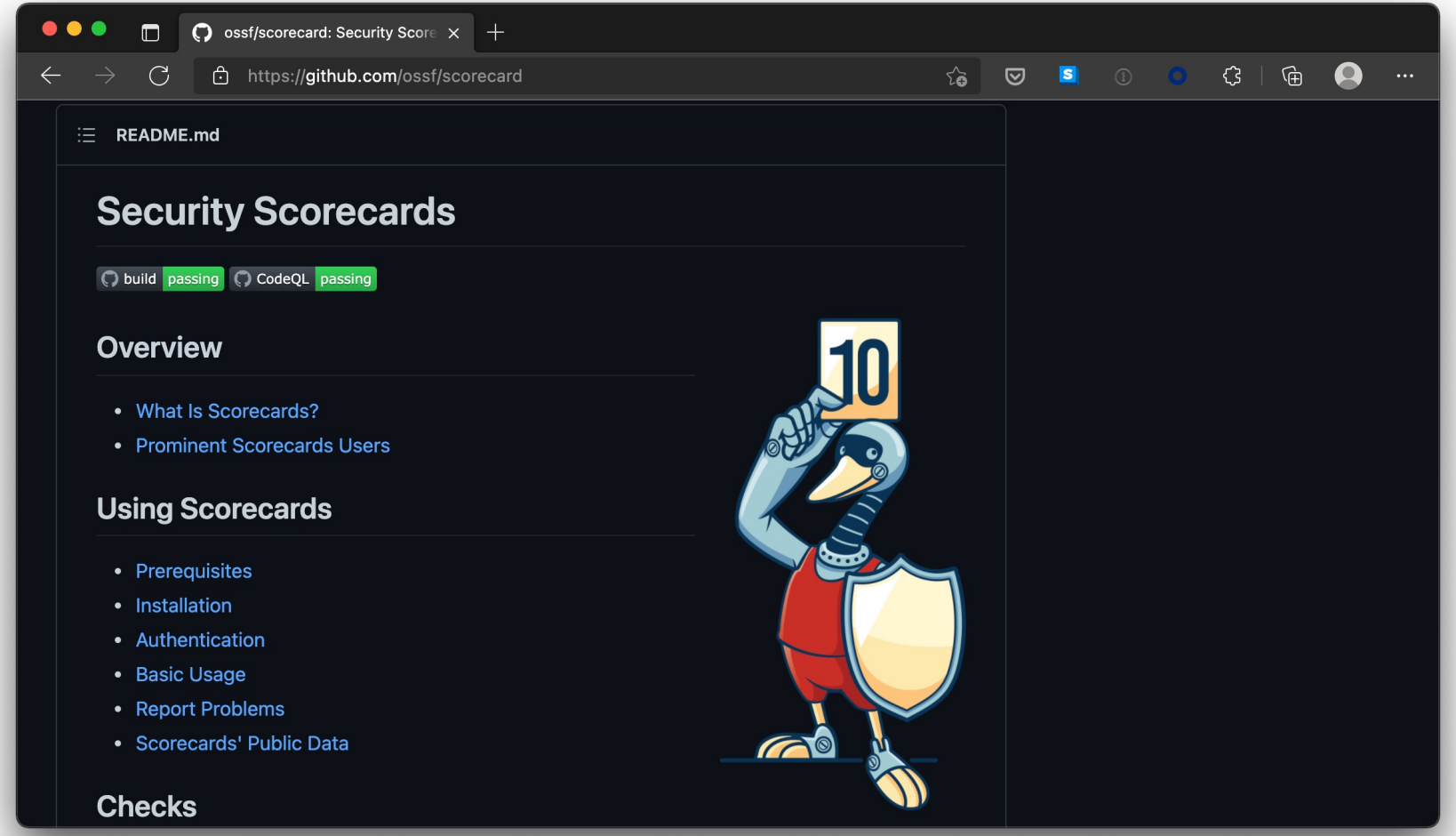
@nielstanis

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

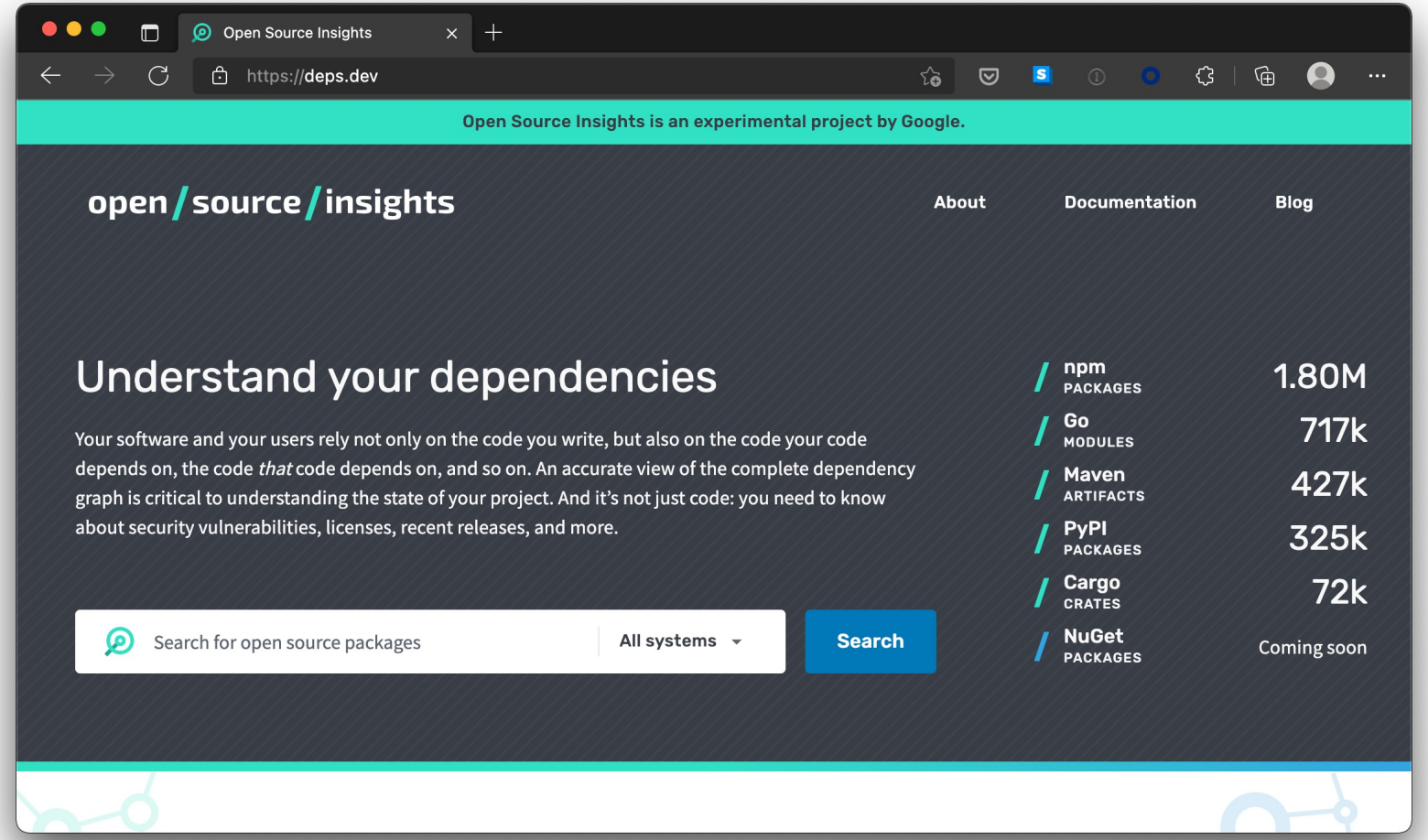
Security Scorecards - OpenSSF



WeAreDevelopers

@nielstanis

Deps.Dev by Google



WeAreDevelopers

@nielstanis

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.

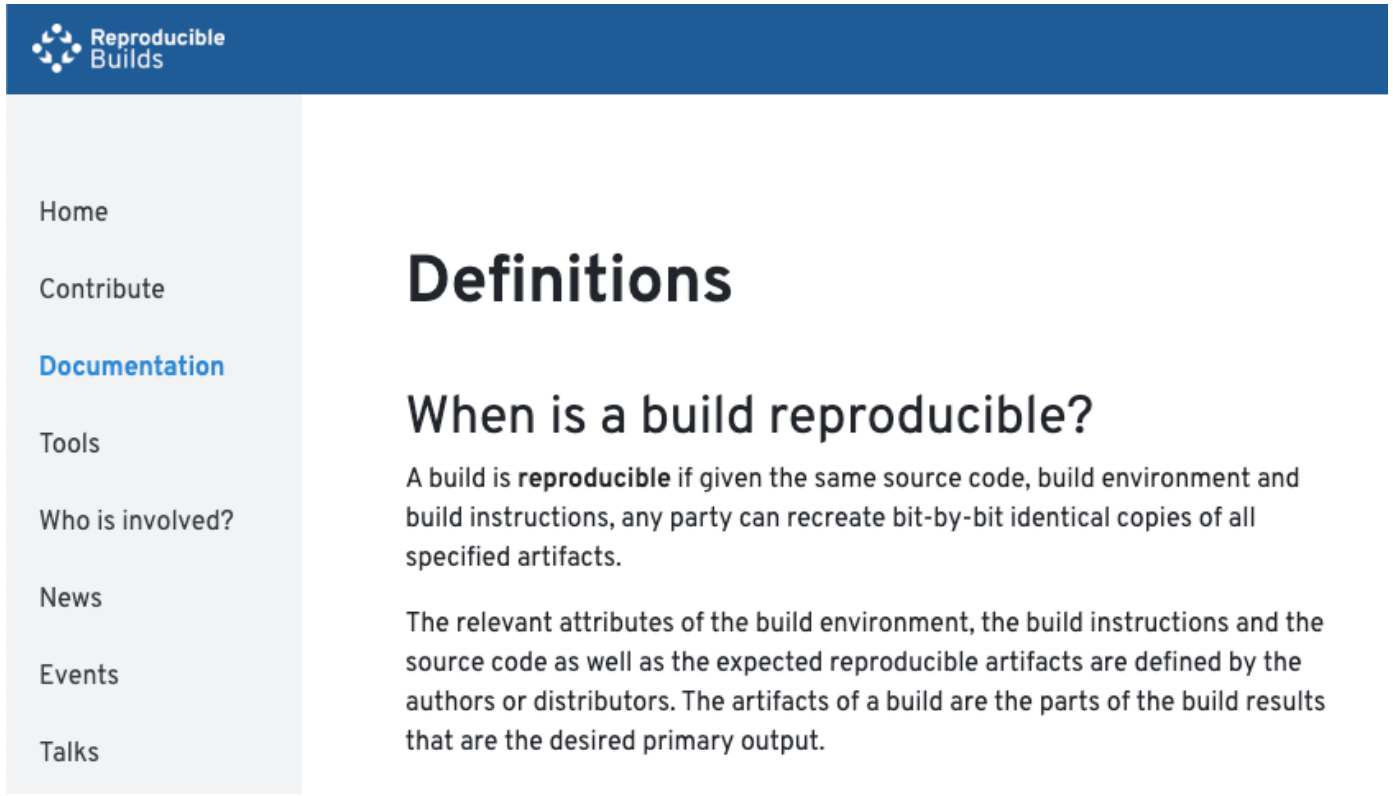
SCORE
5.8/10

Scorecard as of April 25, 2022.

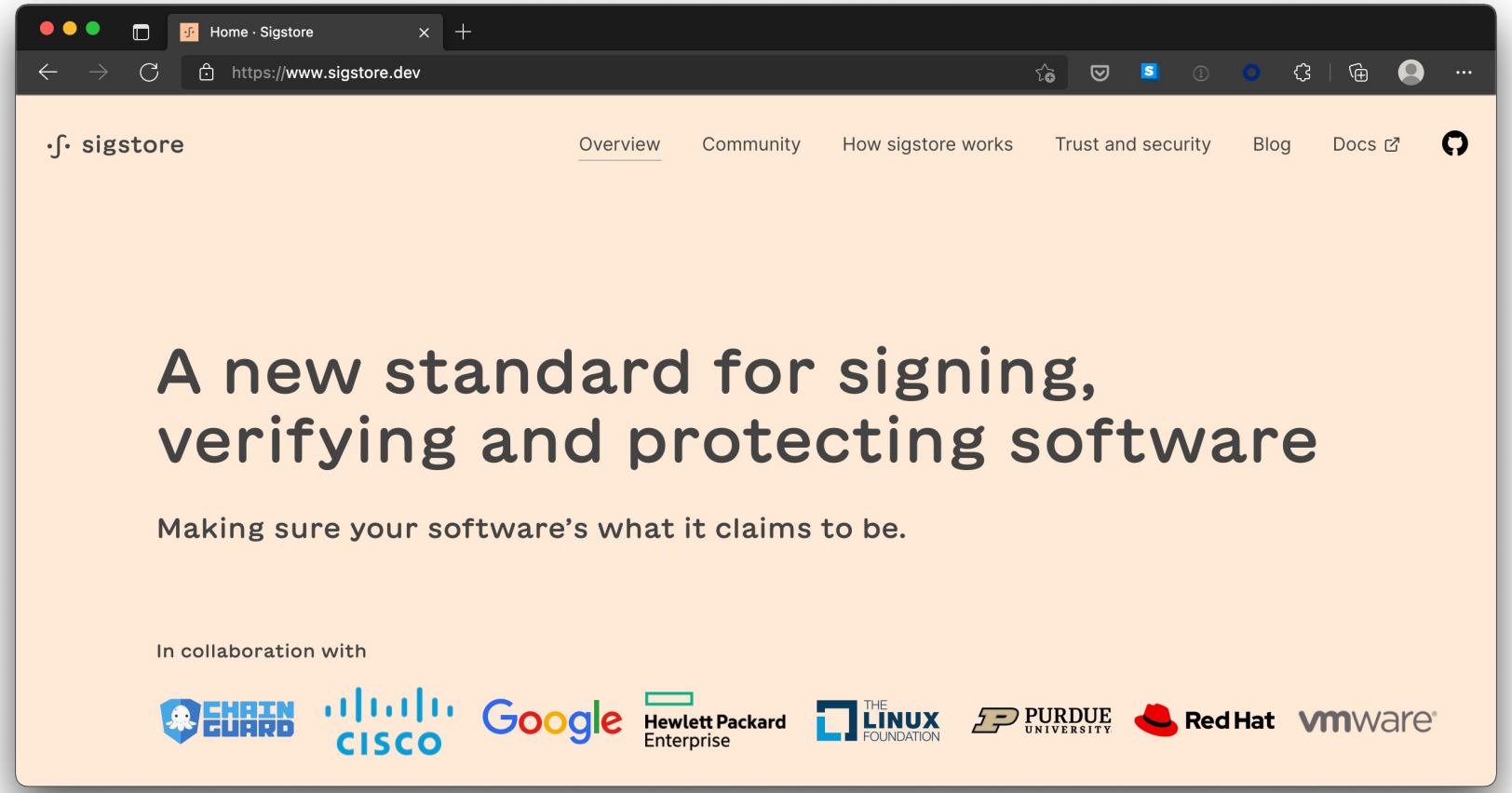
Code-Review	5/10
Maintained	10/10
CII-Best-Practices	0/10
Vulnerabilities	10/10
Dependency-Update-Tool	0/10
Security-Policy	10/10
Dangerous-Workflow	10/10
Token-Permissions	0/10
License	10/10
Pinned-Dependencies	8/10
Binary-Artifacts	10/10
Fuzzing	0/10
Signed-Releases	0/10

Project metadata as of May 7, 2022.

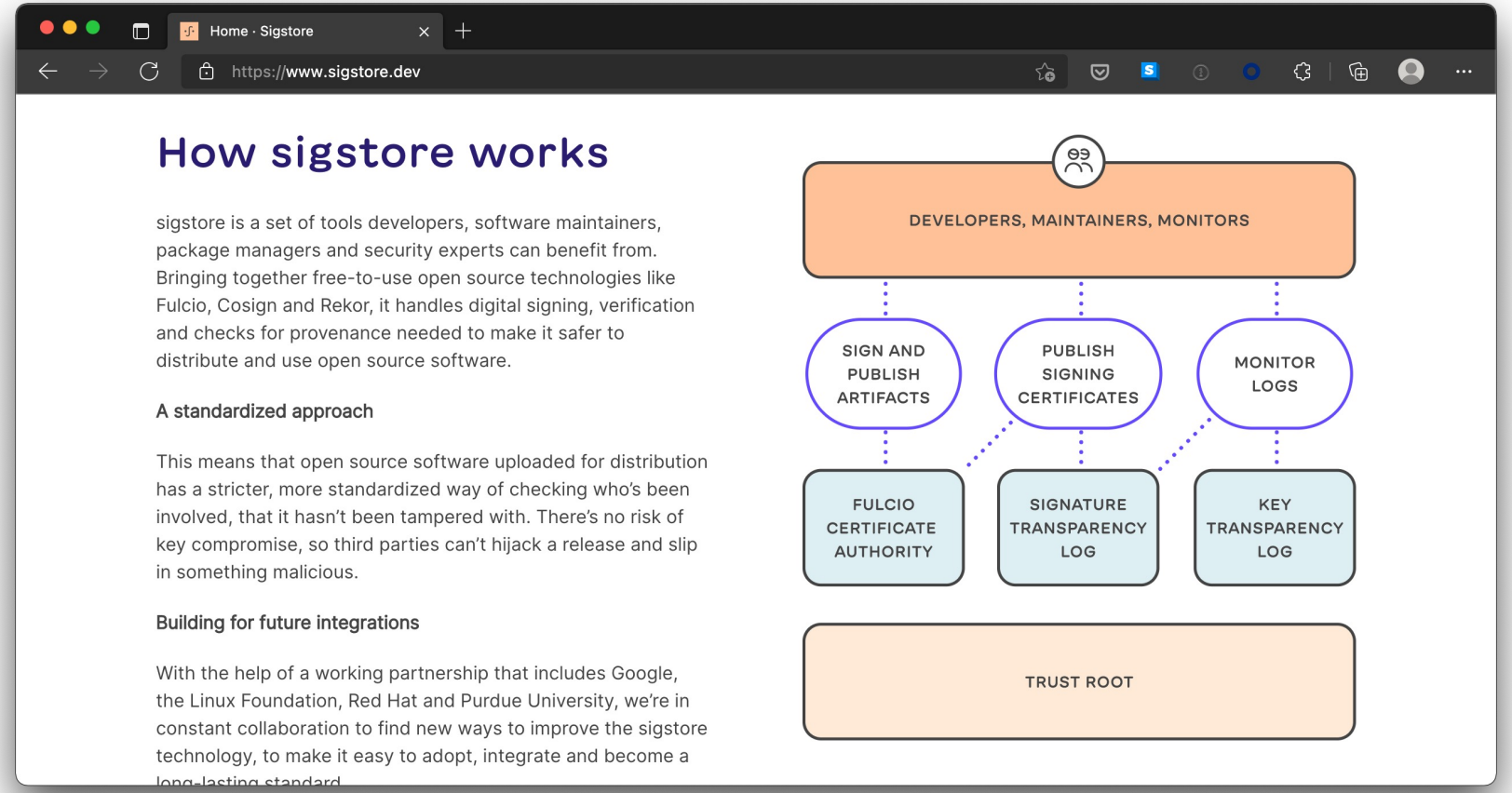
Reproducible/Deterministic Builds



Signing artifacts



Signing artifacts



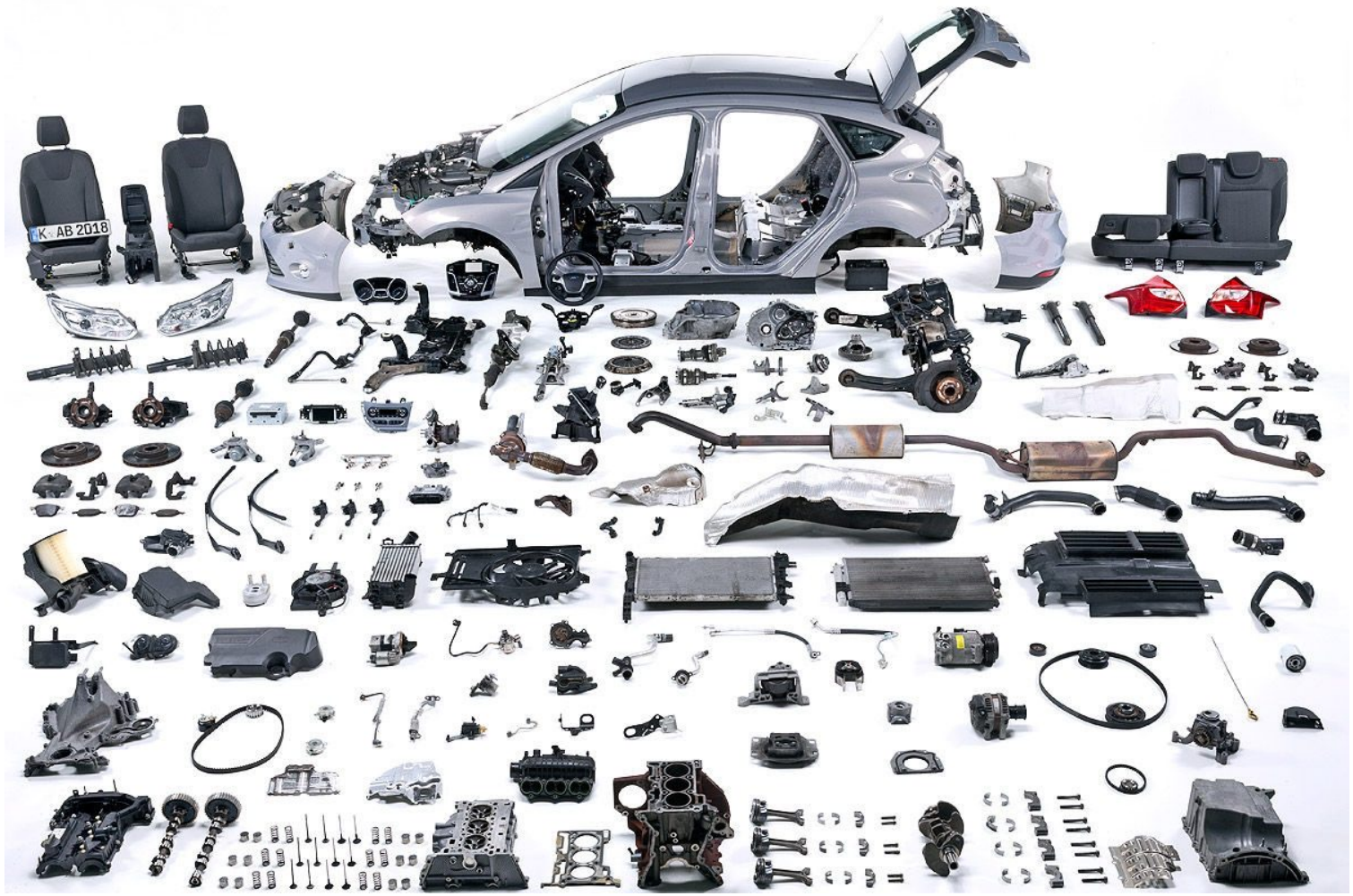
WeAreDevelopers

@nielstanis

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

Automotive Industry



WeAreDevelopers

@nielstanis

Car Supply Chain

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

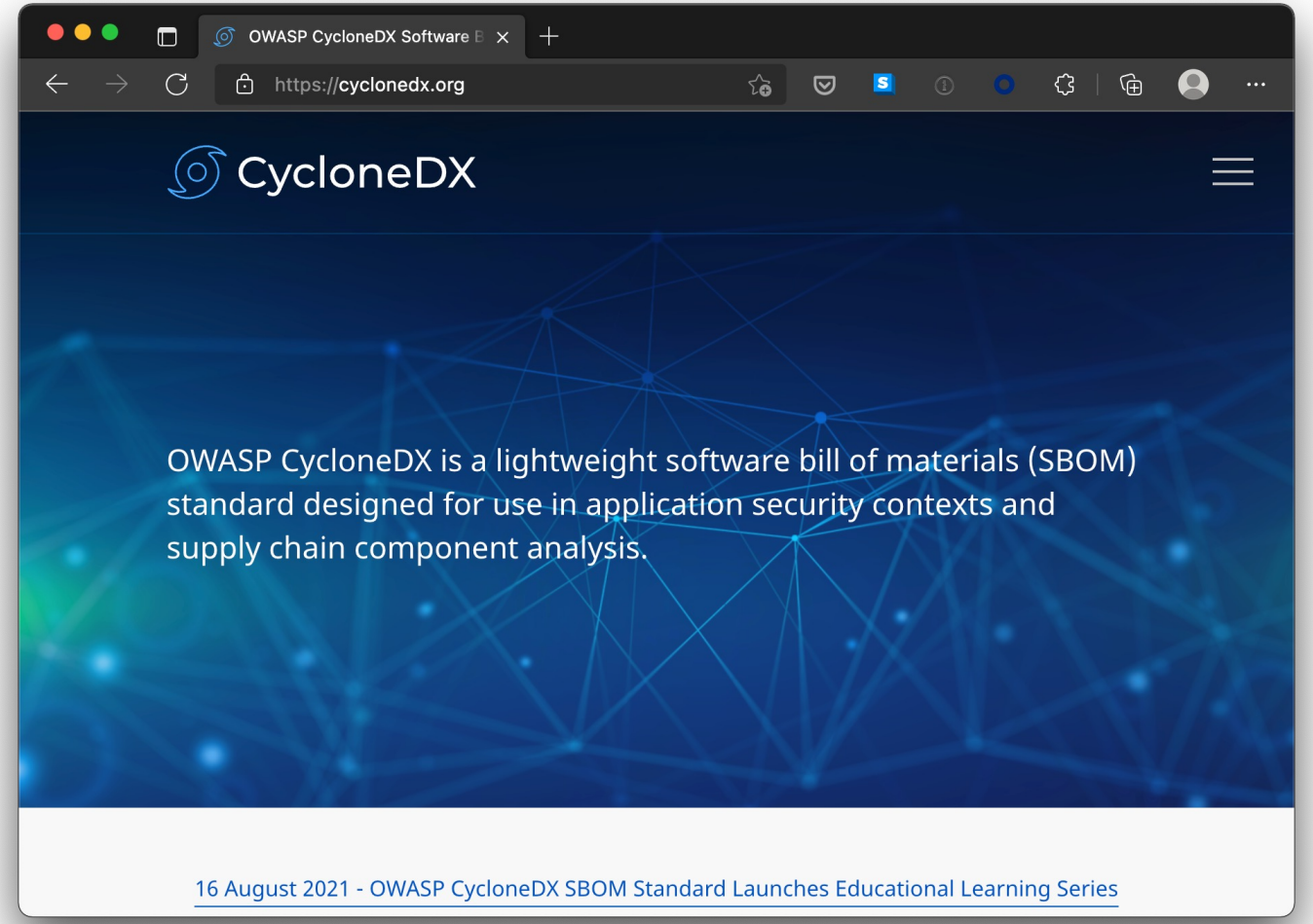
Renault Manufacturing

- 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?

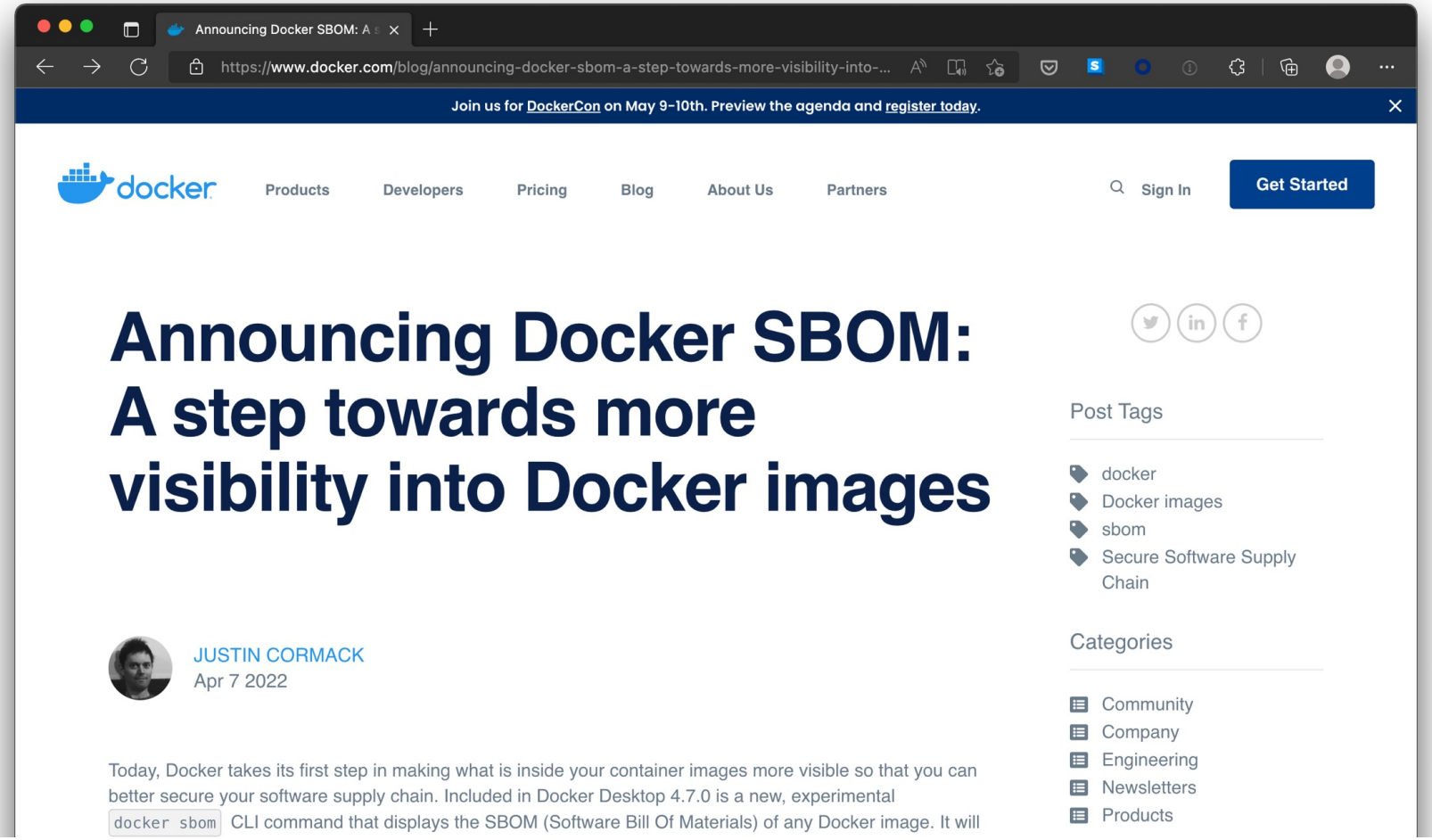
Software Bill of Materials (SBOM)



WeAreDevelopers

@nielstanis

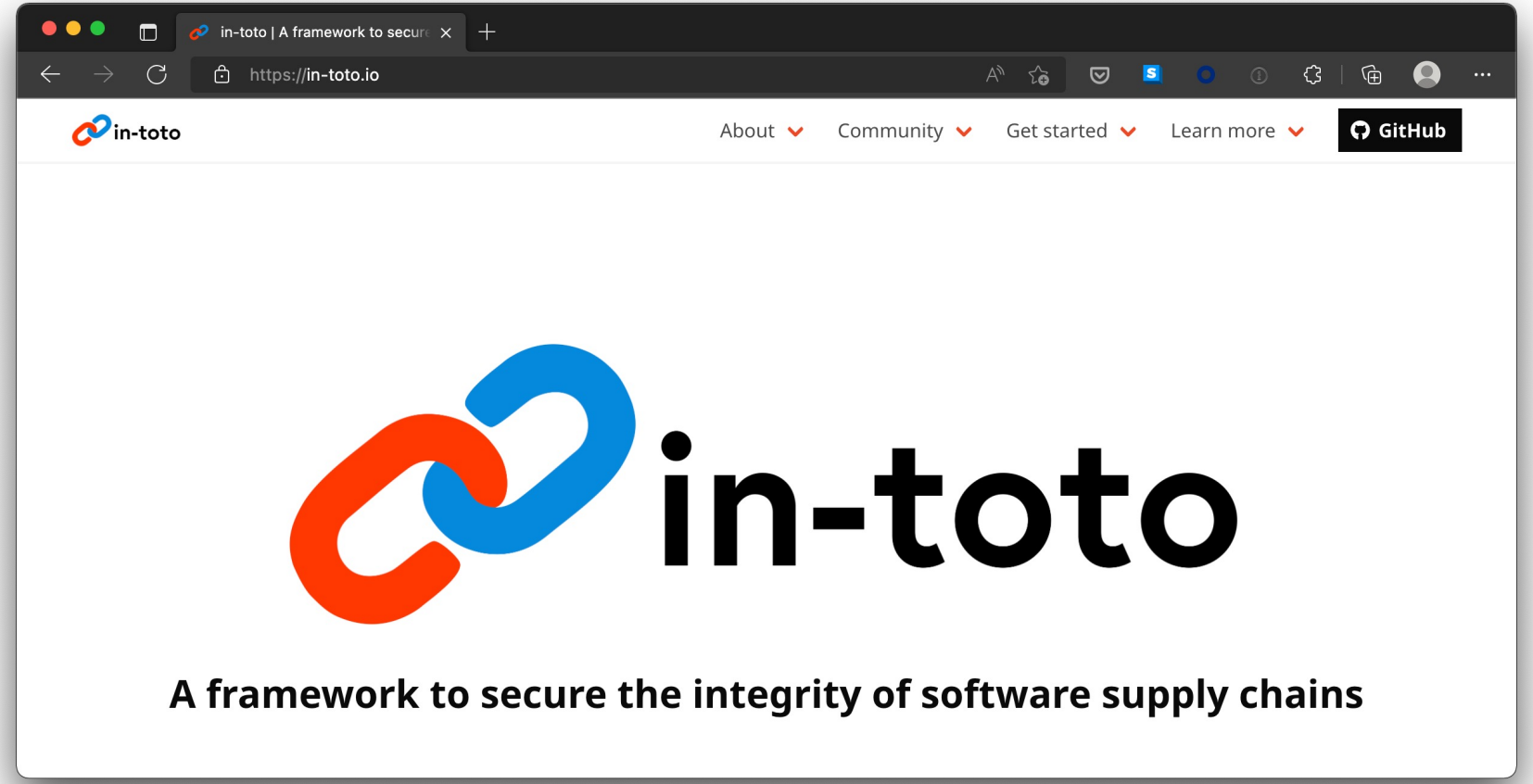
Docker SBOM



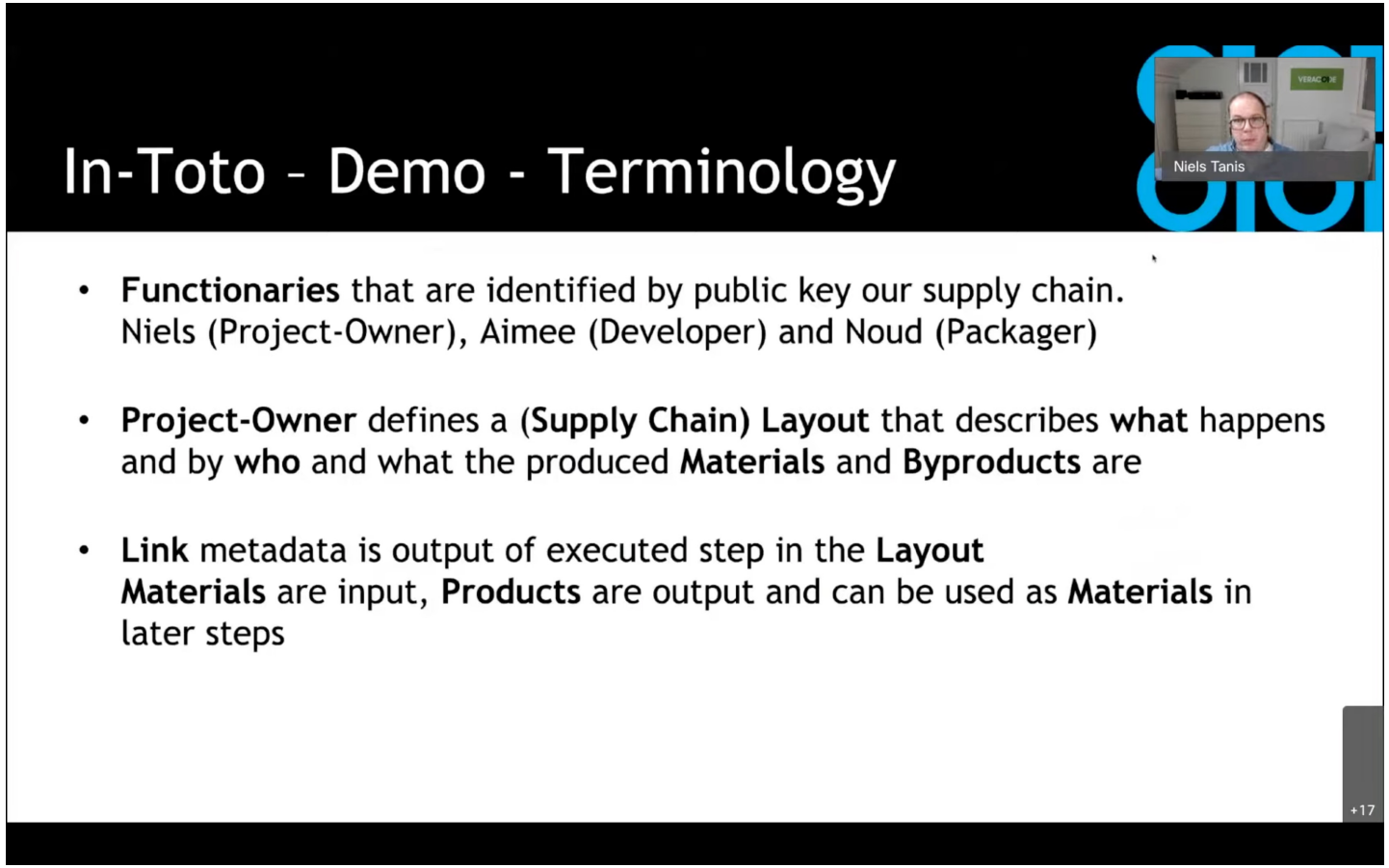
WeAreDevelopers

@nielstanis

In-toto


in-toto | A framework to secure
https://in-toto.io
in-toto
About
Community
Get started
Learn more
GitHub
in-toto
A framework to secure the integrity of software supply chains

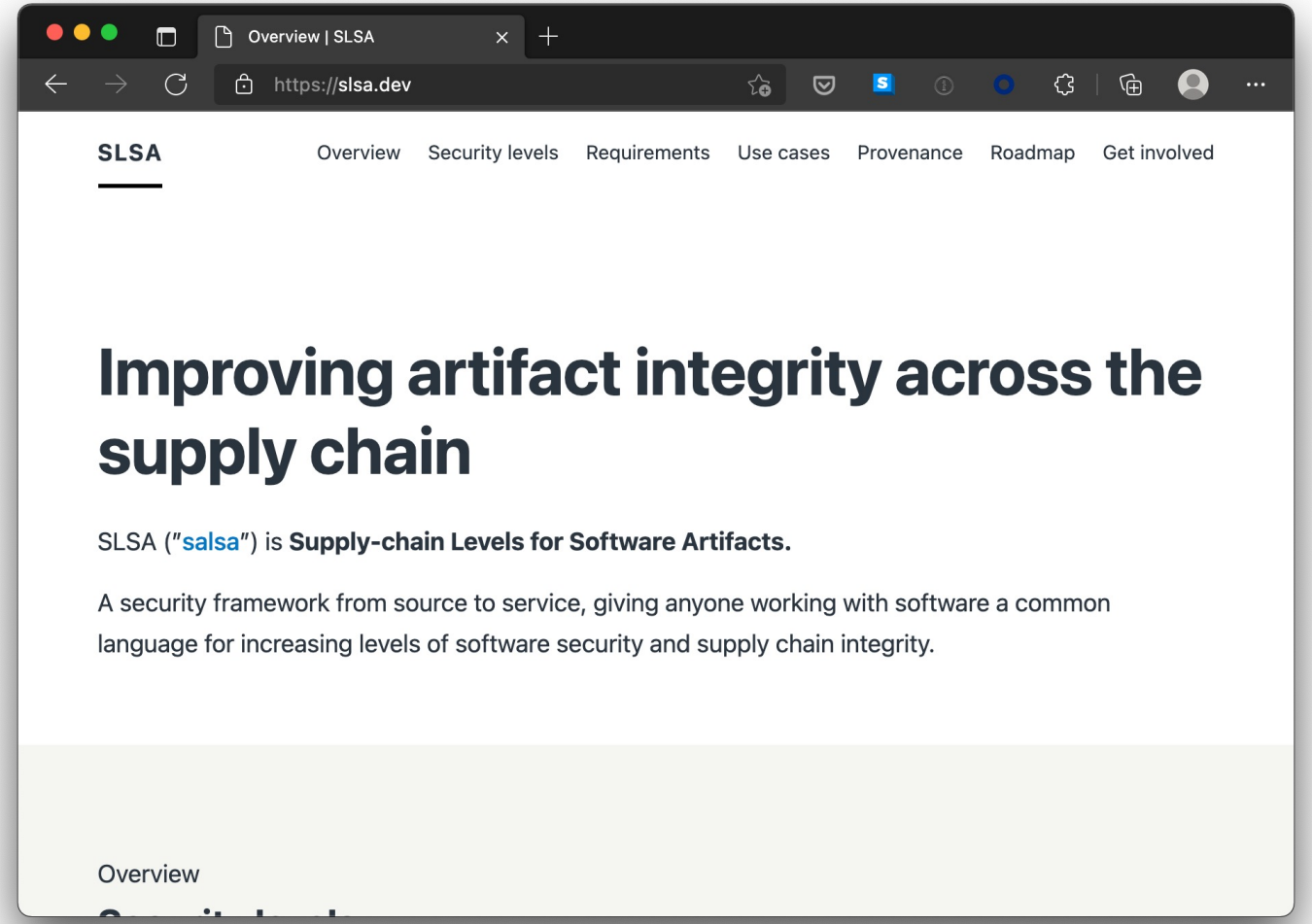
In-Toto - Demo


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

MyAwesomeWebApp Demo

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

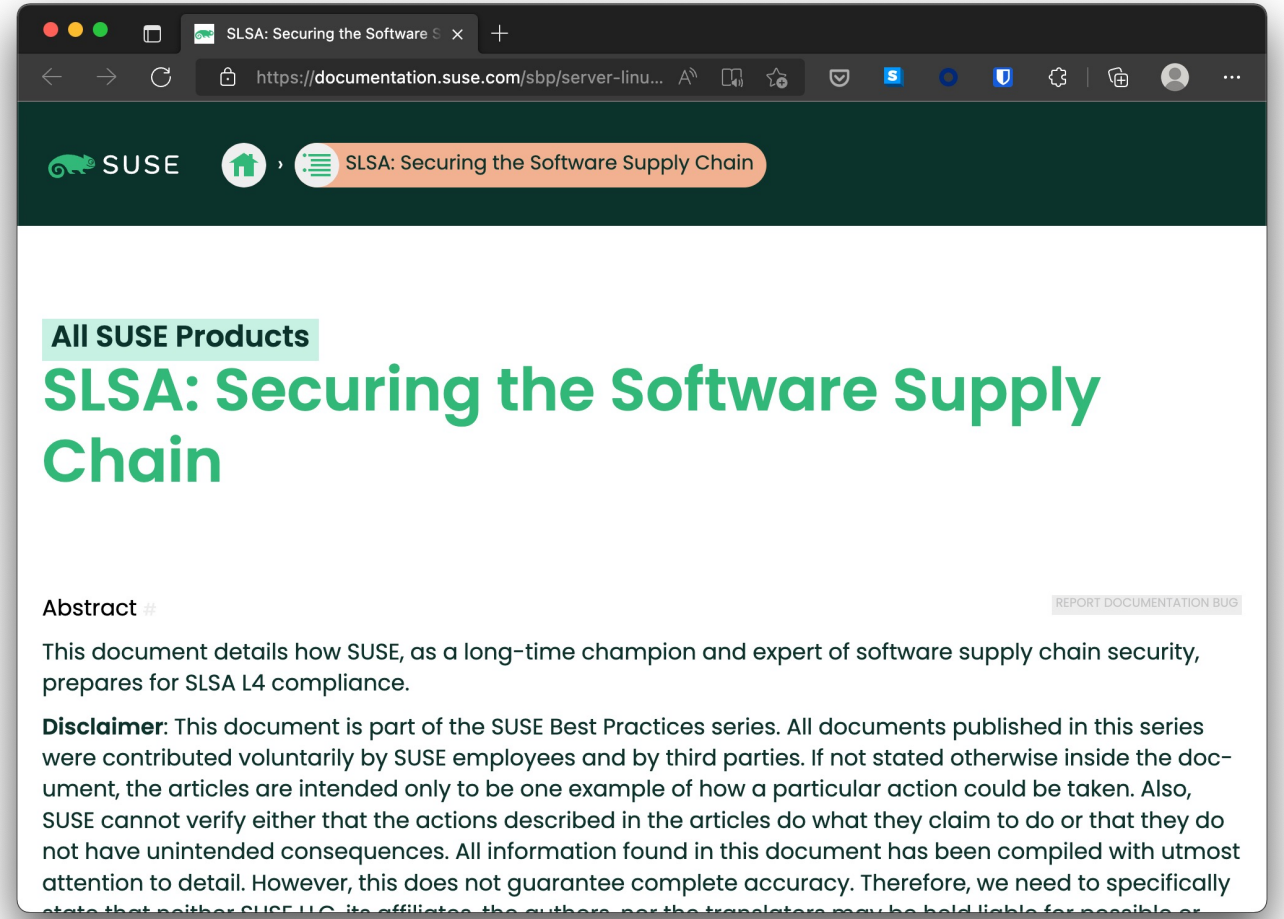
Google SLSA


Overview | SLSA
https://slsa.dev
SLSA
Overview
Security levels
Requirements
Use cases
Provenance
Roadmap
Get involved
Improving artifact integrity across the supply chain
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.
Overview

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

SUSE SLSA Level 4



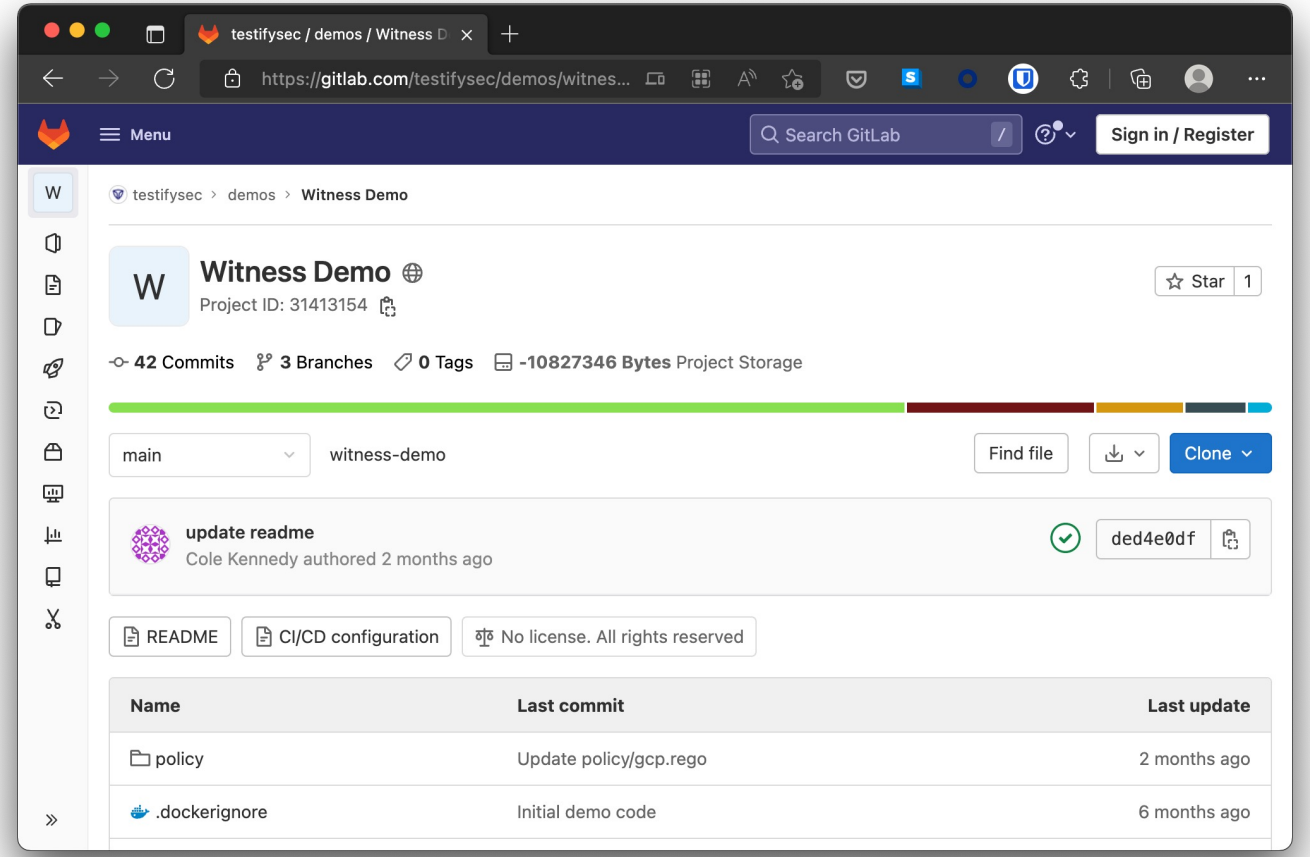
WeAreDevelopers

@nielstanis

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

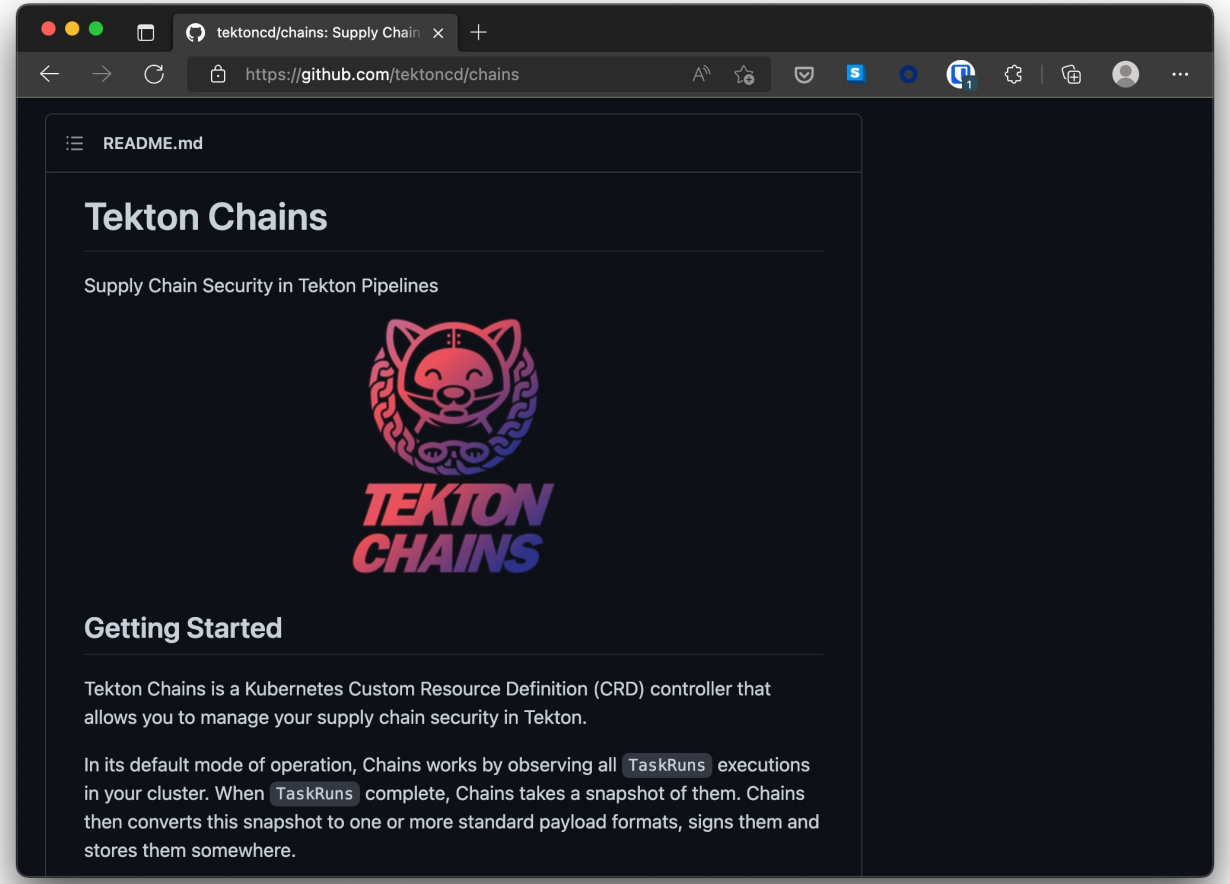
Witness & GitLab Attestator



WeAreDevelopers

@nielstanis

Open Shift - Tekton - Tekton Chains

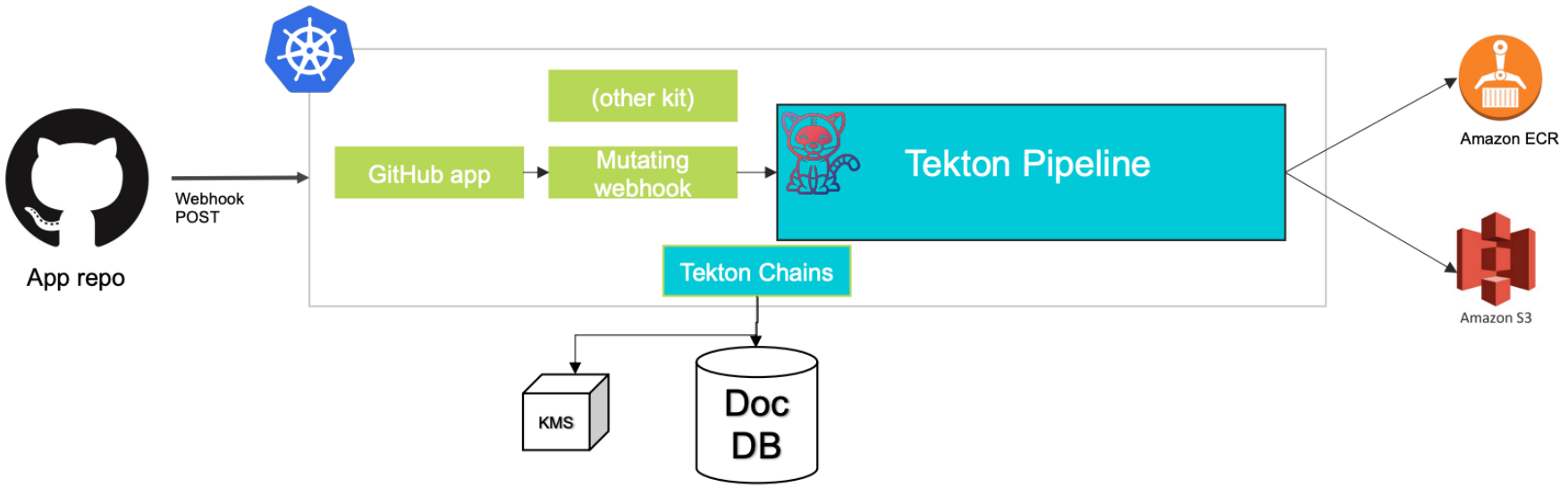


WeAreDevelopers

@nielstanis

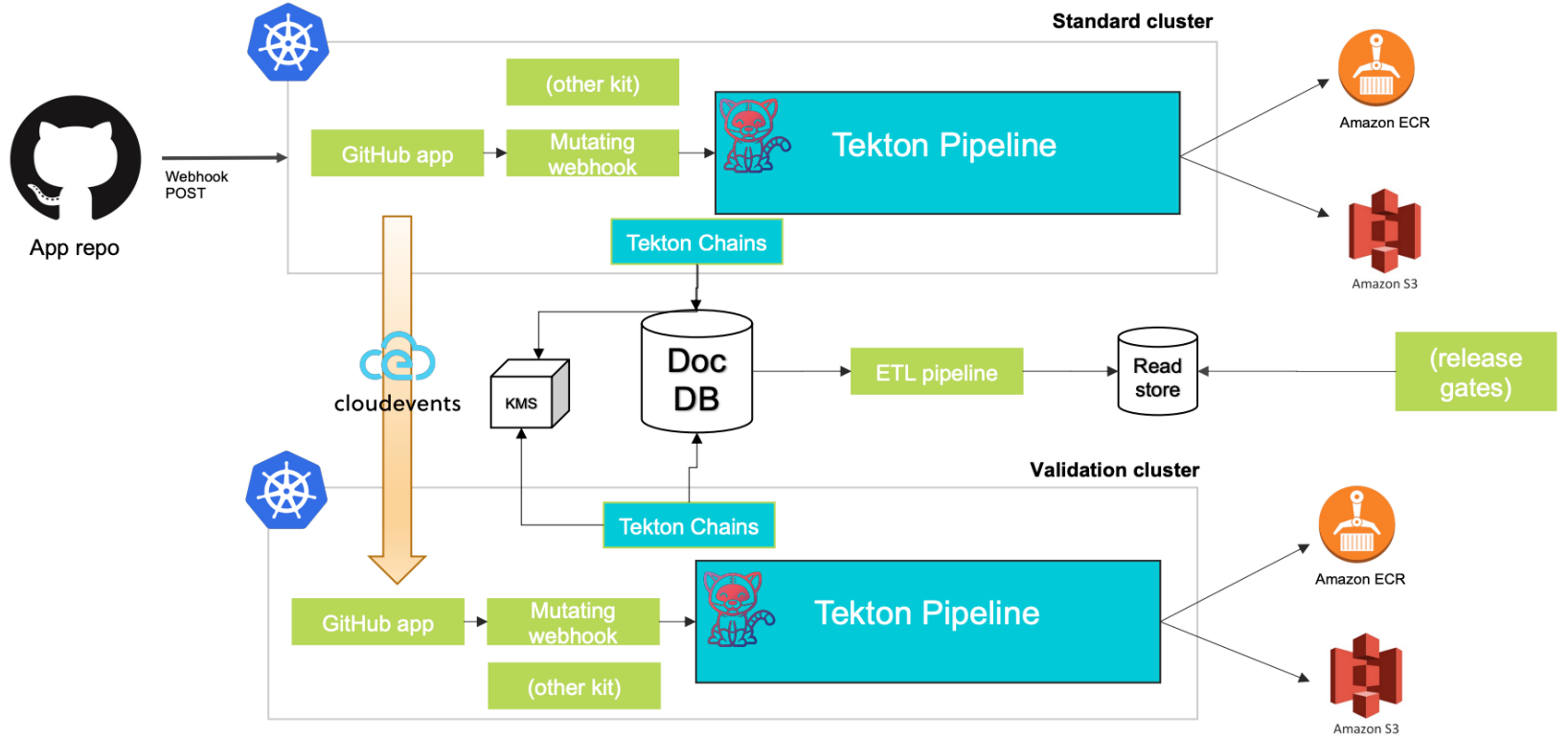
SolarWinds Project Trebuchet

Pipeline With Attestations



SolarWinds Project Trebuchet

Reading Results



WeAreDevelopers

@nielstanis

Working with SBOM

- Kyverno Policy Management
- Chainguard Enfoce
- 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