

NDC { London }

Securing your .NET application
software supply-chain

Niels Tanis

VERACODE



Who am I?

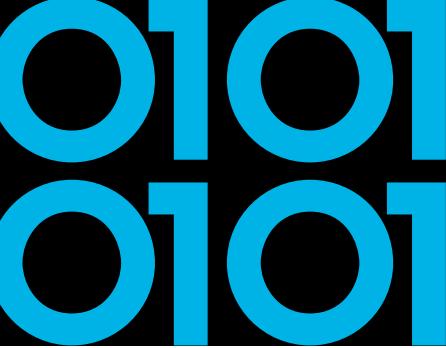
- Niels Tanis
- Principal Security Researcher @ Veracode
 - Background .NET Development, Pentesting/ethical hacking, and software security consultancy
 - ISC² CSSLP
 - Research on static analysis for .NET apps



Securing your .NET application software supply-chain

0101
0101





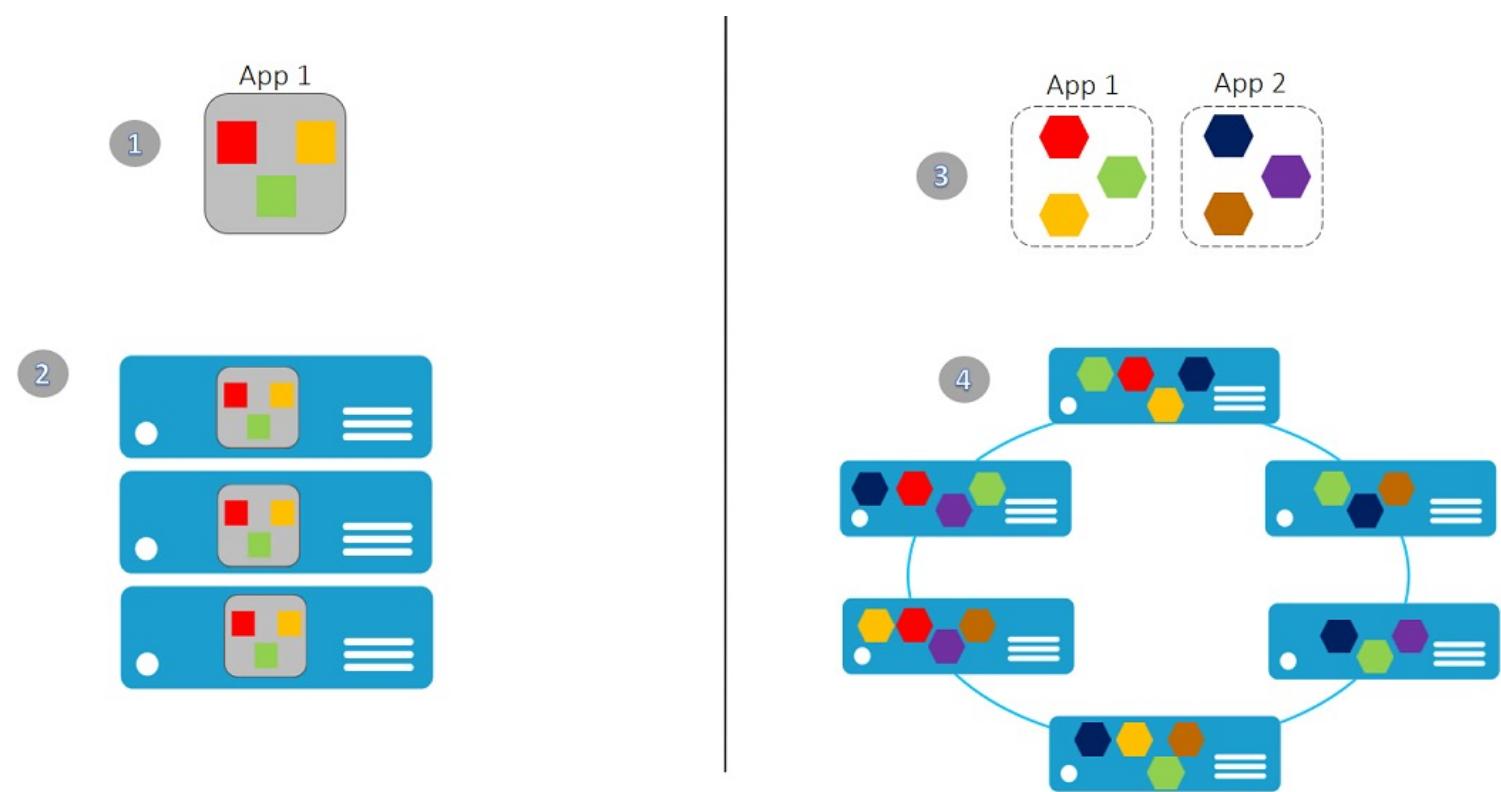
Agenda

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



Evolution in Software Architecture

- Monolith
- Microservices
- Serverless
- Cloud-Native



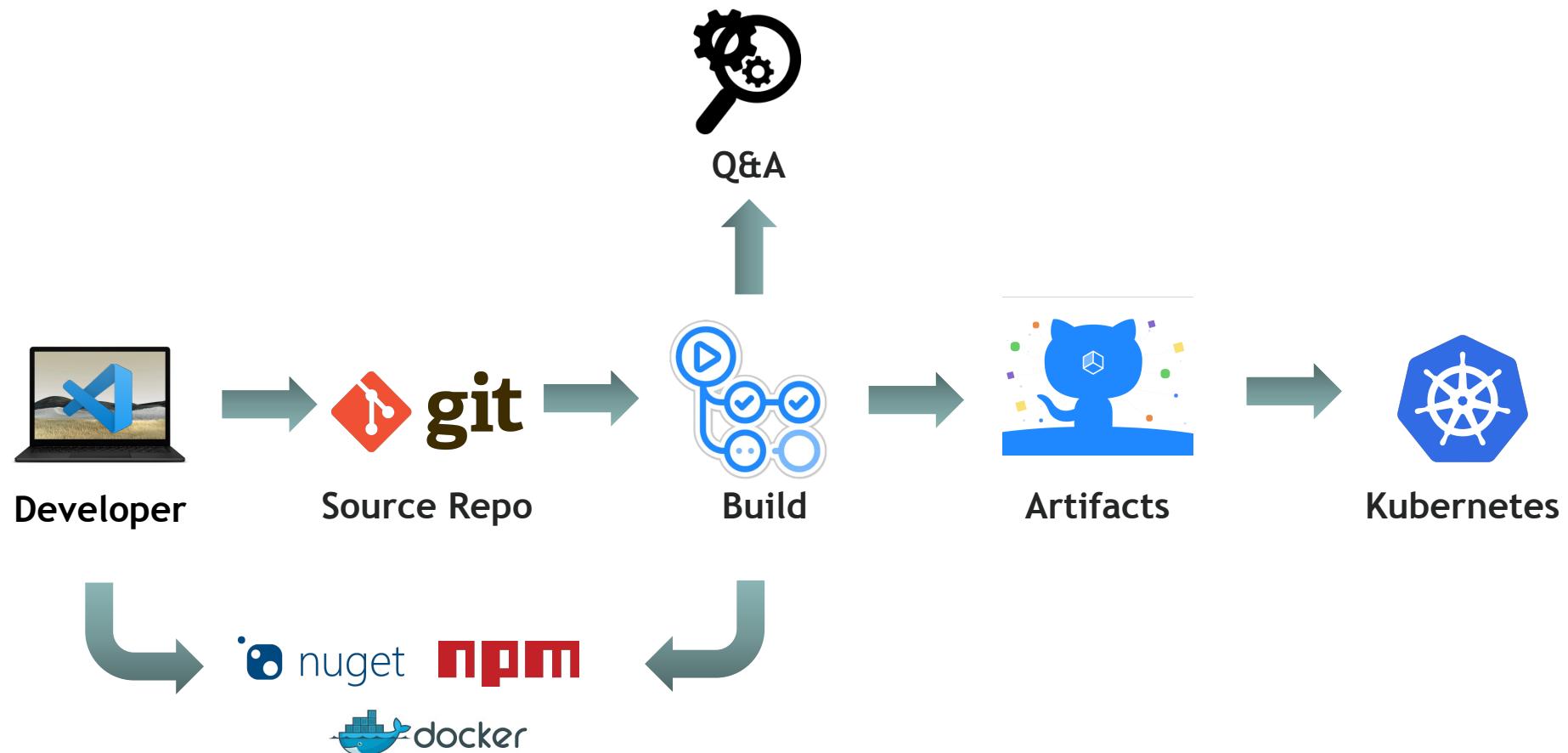
0101
0101

What is a Supply Chain?



Software Supply Chain

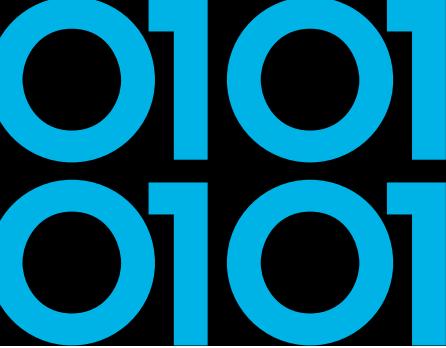
0101
0101



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 orange and red sun-like graphic with radiating lines.



Kaseya

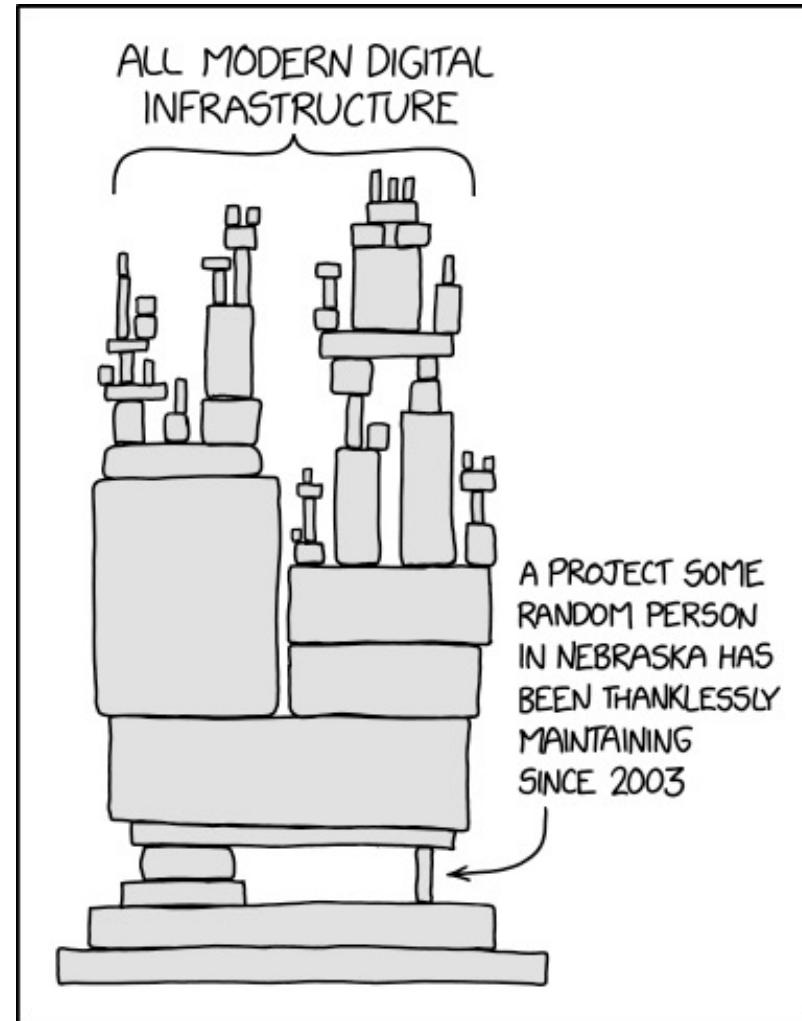
Kaseya Ransomware: a Software Supply Chain Attack or Not?

July 06, 2021 By [Matt Howard](#)



0101
0101

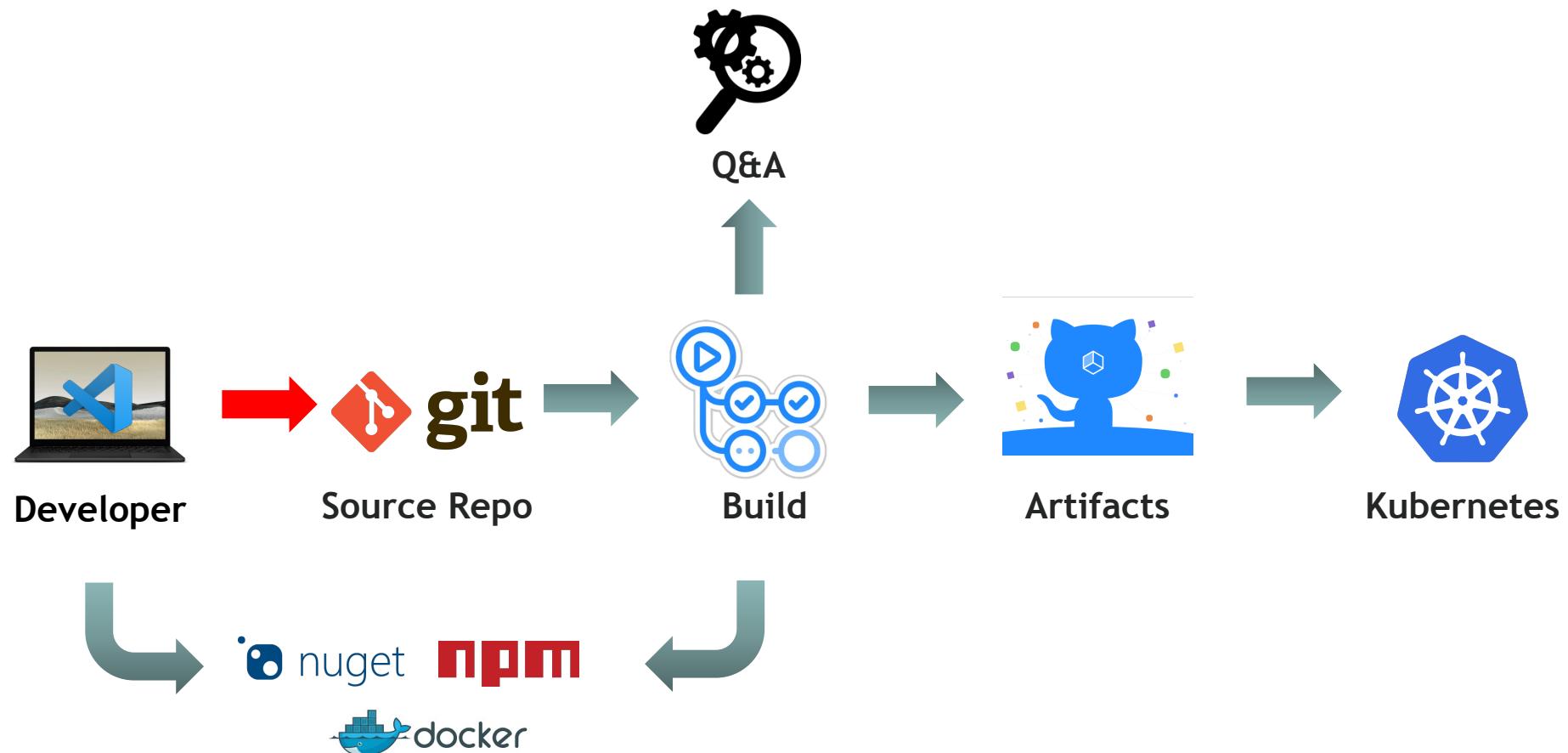
XKDC - Dependency



<https://xkcd.com/2347/>

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 header includes the ZDNet logo, a search bar, and navigation links for AFRICA, UK, ITALY, SPAIN, MORE, EDITION: EU, NEWSLETTERS, ALL WRITERS, and a user profile icon.

A banner at the top of the article reads: "MUST READ: Chinese hackers could steal data now and crack it with quantum computers later, warns report".

Canonical GitHub account hacked, Ubuntu source code safe

Ubuntu source code appears to be safe; however Canonical is investigating.

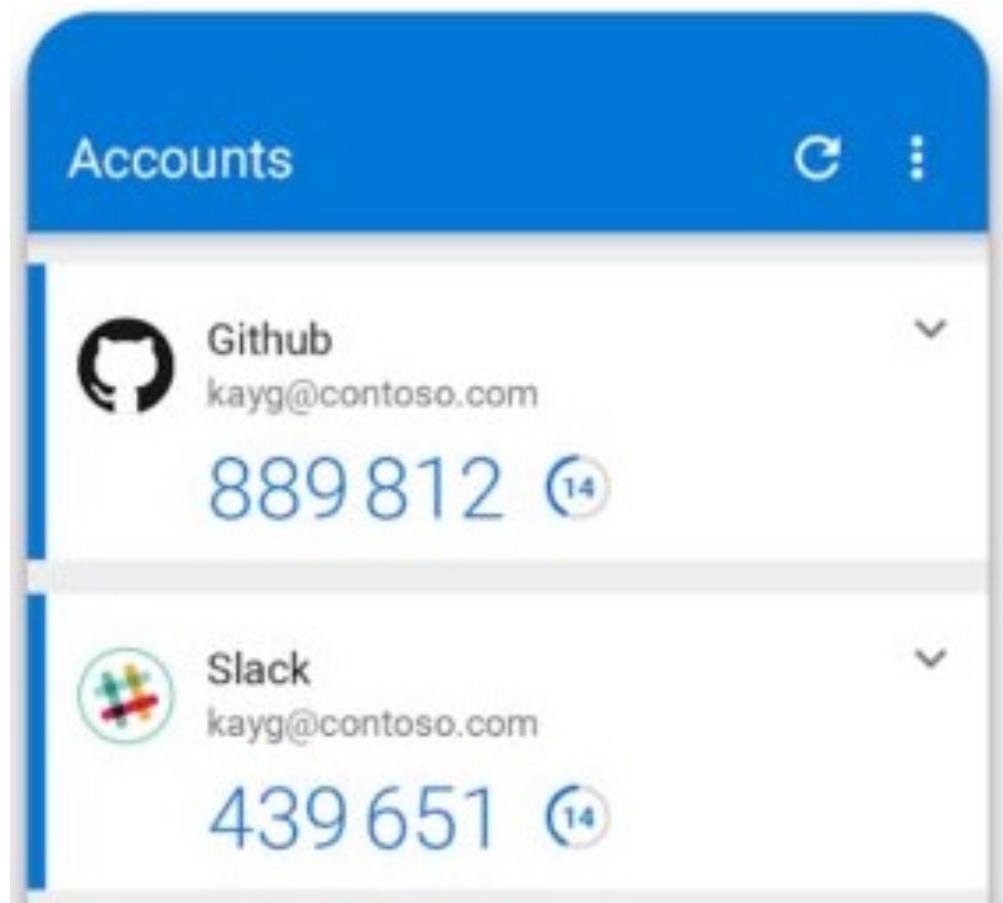
Below the headline, there are social sharing icons for LinkedIn, Facebook, Twitter, and Email, followed by the author's information: "By Catalin Cimpanu for Zero Day | July 7, 2019 | Topic: Security".

The main content area features a screenshot of the Canonical Ltd. GitHub repository page. The repository name is "CAN_GOT_HAXXED_10". The page shows basic repository details like stars, forks, and last update time. To the right of the screenshot, there is a "RELATED" section with a link to another article titled "Why everyone should have this cheap security tool".

At the bottom of the article, a caption reads: "The GitHub account of Canonical Ltd., the company behind the Ubuntu Linux distribution, was hacked earlier this week."

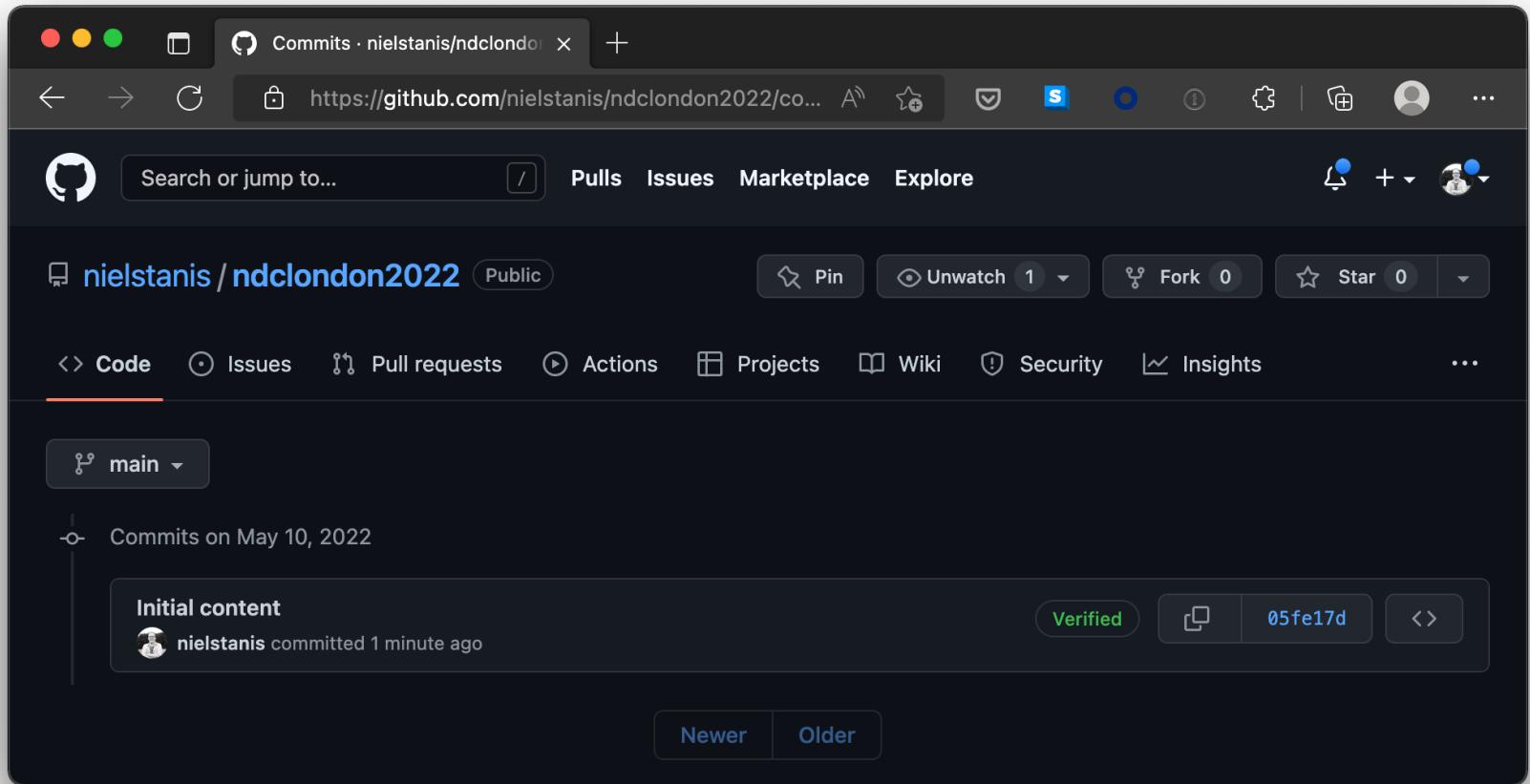
0101
0101

Use MFA on source-repository



0101
0101

GIT Commit Signing



Hypocrite Commits

0101
0101

The screenshot shows a dark-themed web browser window. The address bar displays the URL <https://github.com/QiushiWu/qiushiwu.github.io/blob/main/papers/OpenS.pdf>. The main content area of the browser shows a research paper with the following details:

Title: On the Feasibility of Stealthily Introducing Vulnerabilities in Open-Source Software via Hypocrite Commits

Authors: Qiushi Wu and Kangjie Lu
University of Minnesota
`{wu000273, kjlu}@umn.edu`

Abstract: Open source software (OSS) has thrived since the forming of Open Source Initiative in 1998. A prominent example is the Linux kernel, which has been used by numerous major software vendors and empowering billions of devices. The higher availability and lower costs of OSS boost its adoption, while its openness and flexibility enable quicker innovation. More importantly, the OSS development approach is believed to produce more reliable and higher-quality software since it typically has thousands of independent programmers testing and fixing bugs of the software collaboratively.

In this paper, we instead investigate the insecurity of OSS from a critical perspective—the feasibility of stealthily introducing vulnerabilities in OSS via hypocrite commits (i.e., seemingly legitimate commits that introduce security flaws). We propose a novel attack framework that can automatically identify and exploit such vulnerabilities in OSS. Our experiments show that our framework can successfully identify and exploit various types of vulnerabilities in real-world OSS projects, including the Linux kernel, Apache, and MySQL. This work highlights the potential risks of OSS and calls for more rigorous security analysis and defense mechanisms.

Its openness also encourages contributors; OSS typically has thousands of independent programmers testing and fixing bugs of the software. Such an open and collaborative development not only allows higher flexibility, transparency, and quicker evolution, but is also believed to provide higher reliability and security [21].

A prominent example of OSS is the Linux kernel, which is one of the largest open-source projects—more than 28 million lines of code used by billions of devices. The Linux kernel involves more than 22K contributors. Any person or company can contribute to its development, e.g., submitting a patch



Octopus Scanner - NetBeans

May 28, 2020

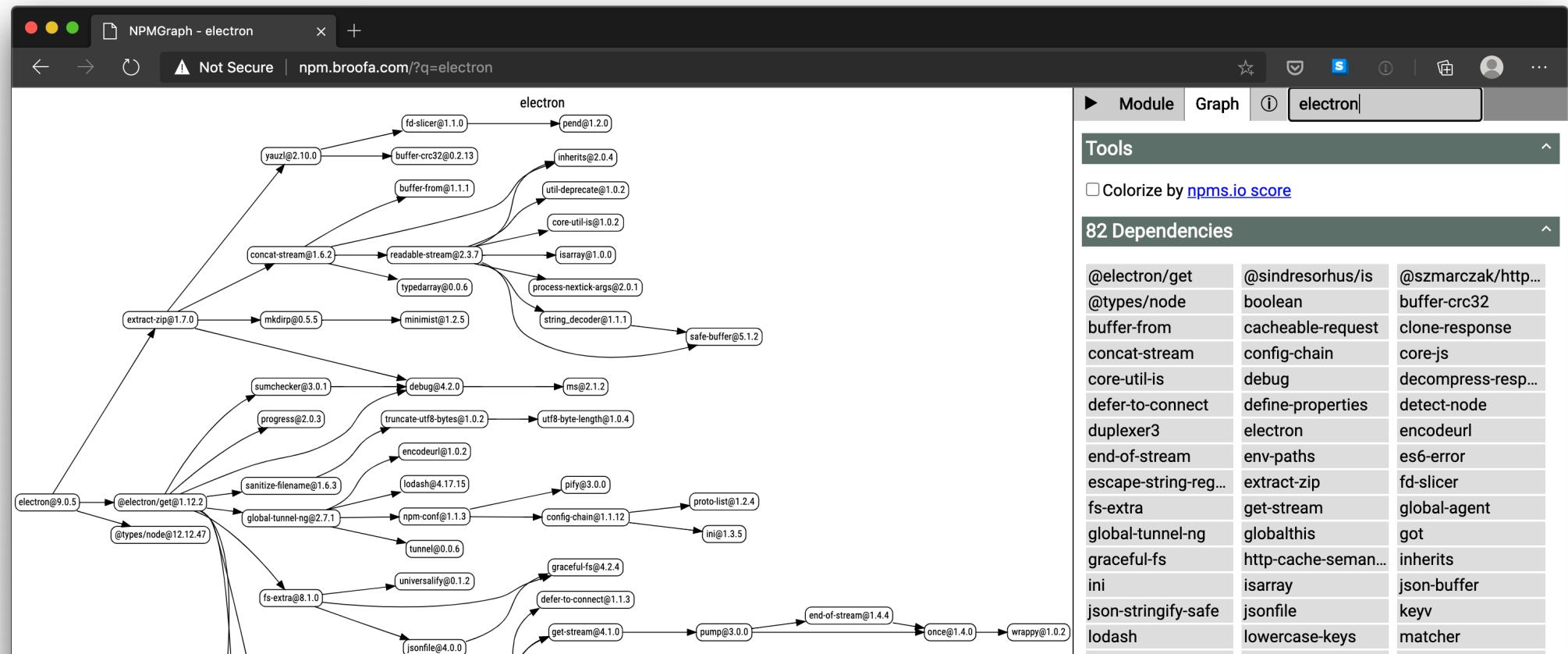
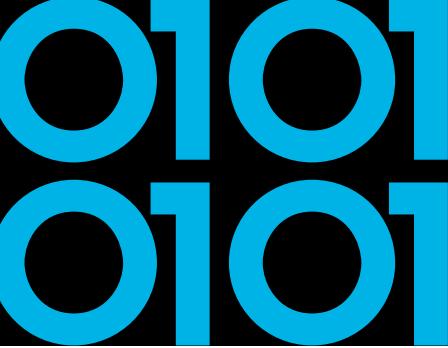
The Octopus Scanner Malware: Attacking the open source supply chain



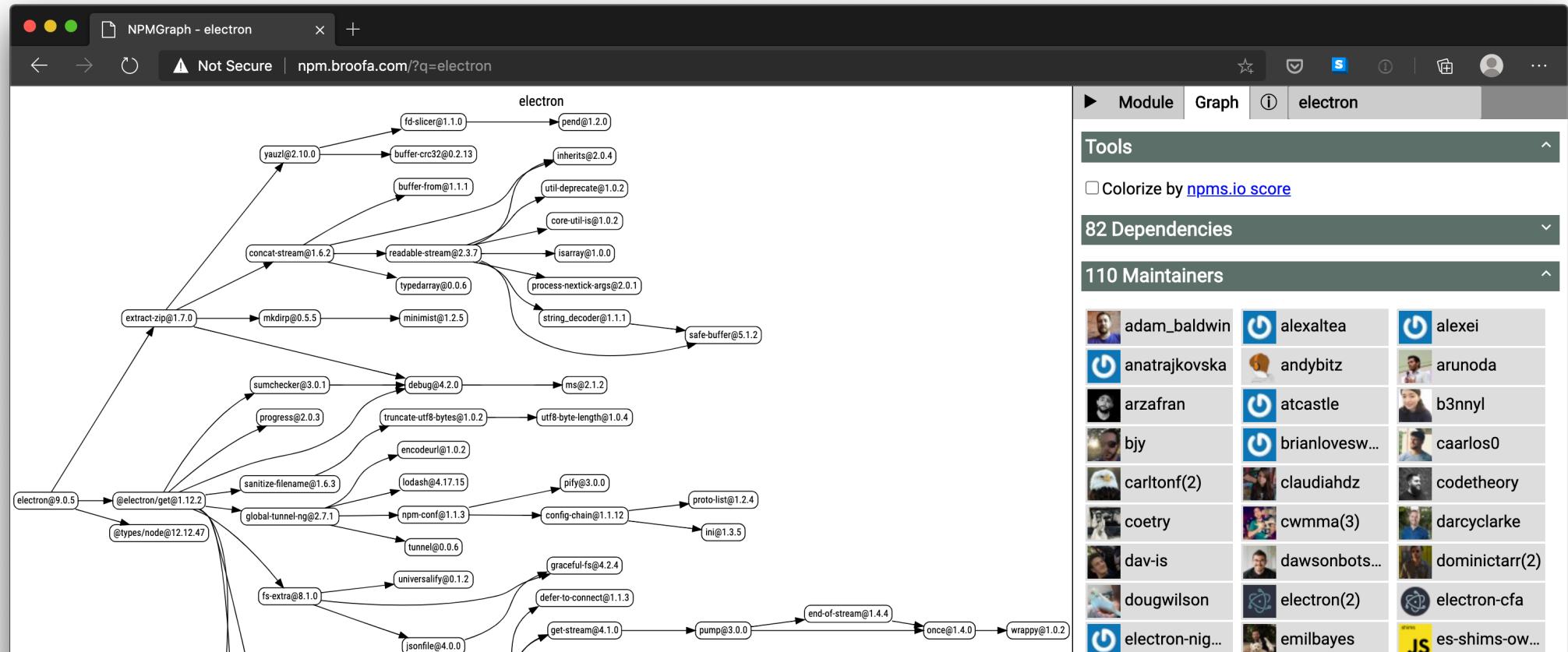
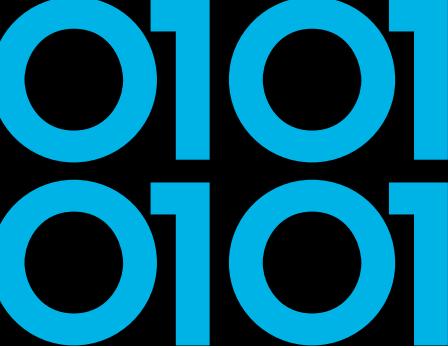
Alvaro Muñoz

Securing the open source supply chain is an enormous task. It goes far beyond a security assessment or just patching for the latest CVEs. Supply chain security is about the integrity of the entire software development and delivery ecosystem. From the code commits themselves, to how they flow through the CI/CD pipeline, to the actual delivery of releases, there's the potential for loss of integrity and security concerns, throughout the entire lifecycle.

Visual Studio Code



Visual Studio Code



0101
0101

Visual Studio Code

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

- Title Bar:** CVE-2022-21991 - Security Up... (partially visible)
- Address Bar:** https://msrc.microsoft.com/update-guide/vulnerability/CVE-2022-21991
- Header:** Microsoft | MSRC | Security Updates | Acknowledgements | Developer
- Breadcrumbs:** MSRC > Customer Guidance > Security Update Guide > Vulnerabilities > CVE 2022 21991
- Message:** Welcome to the new and improved Security Update Guide! We'd love your feedback. Please click here to share your thoughts or email us at msrc_eng_support@microsoft.com. Thank you!
- Section Title:** Visual Studio Code Remote Development Extension Remote Code Execution Vulnerability
- CVE ID:** CVE-2022-21991
- On this page:** Security Vulnerability
- Released:** Feb 8, 2022
- Assigning CNA:** Microsoft
- Links:** CVE-2022-21991, CVSS:3.1 8.1 / 7.1

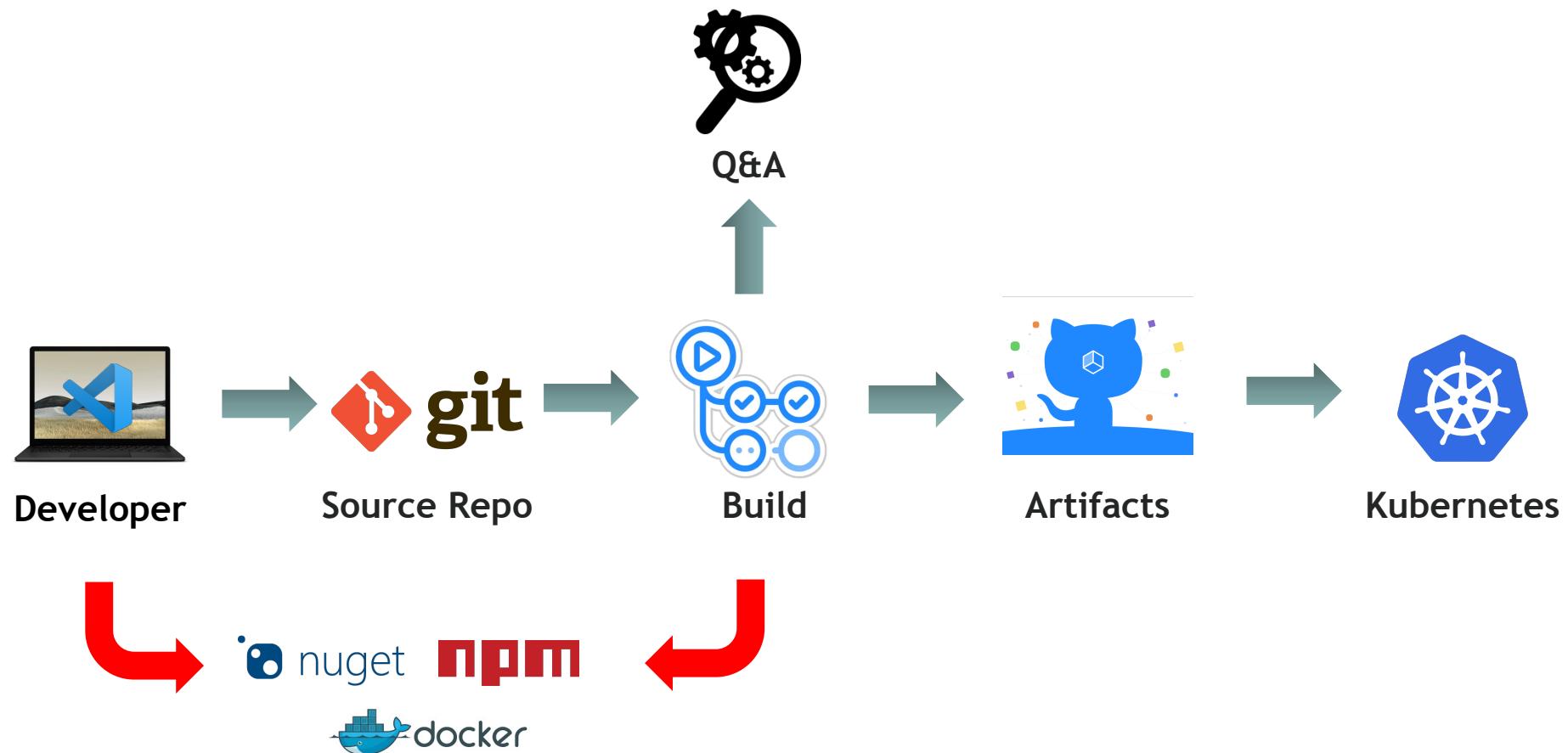
0101
0101

Visual Studio Code



3rd Party Libraries

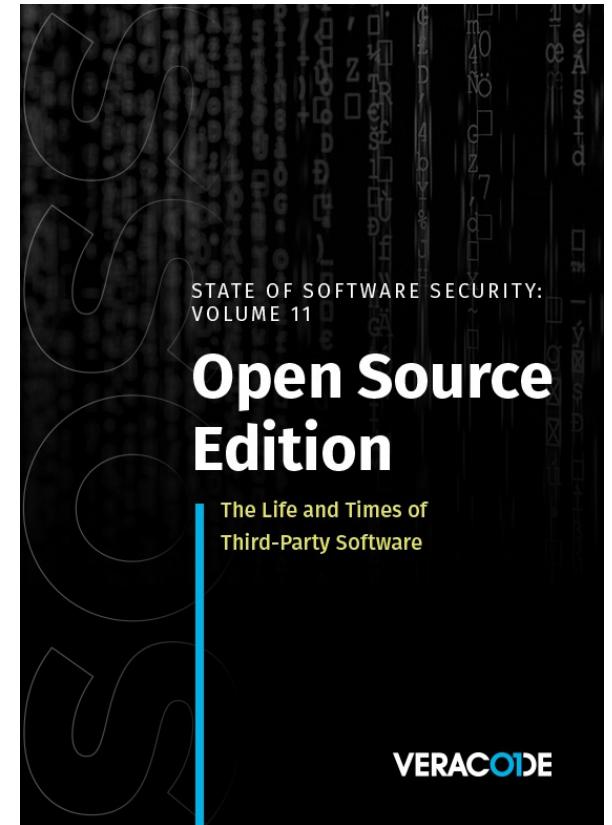
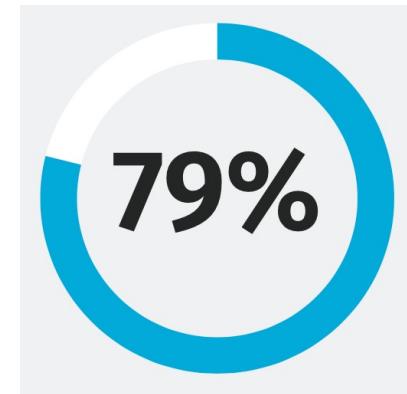
0101
0101



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.”*



0101
0101

Vulnerabilities in libraries

The screenshot shows a GitHub issue page for the repository `dotnet/announcements`. The issue is titled "Microsoft Security Advisory CVE-2022-24512 | .NET Remote Code Execution Vulnerability #213". The issue was opened by `dcwhittaker` on March 8th, 2022, with 0 comments. The issue details section includes a comment from `dcwhittaker` providing an executive summary of the vulnerability, stating that Microsoft is releasing a security advisory for a .NET Remote Code Execution vulnerability in .NET 6.0, .NET 5.0, and .NET Core 3.1. The advisory also provides guidance on how to update applications. The issue is labeled with "Monthly-Update", ".NET Core 3.1", ".NET 5.0", ".NET 6.0", "Patch-Tuesday", and "Security". There are no assignees or projects assigned to this issue.

Search or jump to... / Pulls Issues Marketplace Explore

dotnet / announcements Public Unwatch 3k Fork 39 Star 965

Code Issues 186 Pull requests Security Insights

Microsoft Security Advisory CVE-2022-24512 | .NET Remote Code Execution Vulnerability #213

Open dcwhittaker opened this issue on 8 Mar · 0 comments

dcwhittaker commented on 8 Mar · edited Member ...

Microsoft Security Advisory CVE-2022-24512 | .NET Remote Code Execution Vulnerability

Executive summary

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 their applications to remove this vulnerability. A Remote Code Execution vulnerability exists in .NET 6.0, .NET 5.0, and .NET Core 3.1 where a stack buffer overrun occurs in .NET Double Parse routine.

Discussion

Discussion for this issue can be found at [dotnet/runtime#66348](#)

Assignees
No one assigned

Labels
Monthly-Update, .NET Core 3.1, .NET 5.0, .NET 6.0, Patch-Tuesday, Security

Projects
None yet

Milestone
No milestone

Vulnerabilities in libraries

0101
0101



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](#).

Vulnerabilities in libraries



A screenshot of a web browser showing a blog post from GitHub. The title is "GitHub's commitment to npm ecosystem security". It features a photo of Mike Hanley and a summary about the npm registry's security and transparency. A call-to-action at the bottom encourages reading more about automated malware detection and 2FA.

November 15, 2021 — Open Source, Security

GitHub's commitment to npm ecosystem security

Mike Hanley

The npm registry is central to all JavaScript development, and, as stewards of the registry, ensuring its security is a responsibility GitHub takes seriously. Transparency is key in maintaining the trust of our community. Today, we are sharing details of recent incidents on the npm registry, the details of our investigations, and how we're continuing to invest in the security of npm. These investments include the requirement of two-factor authentication (2FA) during authentication for maintainers and admins of popular packages on npm, starting with a cohort of top packages in the first quarter of 2022.

Read on to learn more.

Automated malware detection and requiring 2FA to combat maintainer account takeovers

A screenshot of a web browser showing another GitHub blog post. The title is "Enrolling all npm publishers in enhanced login verification and next steps for two-factor authentication enforcement". It announces a staged rollout of enhanced login verification for npm publishers starting December 7th. The GitHub logo is visible in the footer.

Blog Engineering Community Product Security Open Source Enterprise More + Q

Open Source Security

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

Today we're introducing enhanced login verification to the npm registry, and we will begin a staged rollout to maintainers beginning Dec 7.

0101
0101

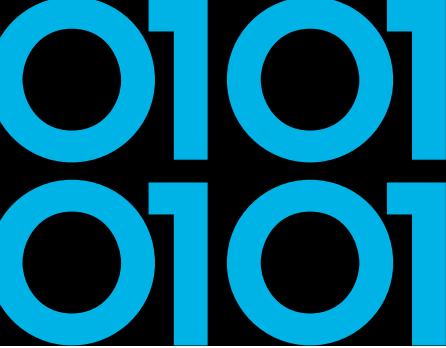
Dependency Confusion





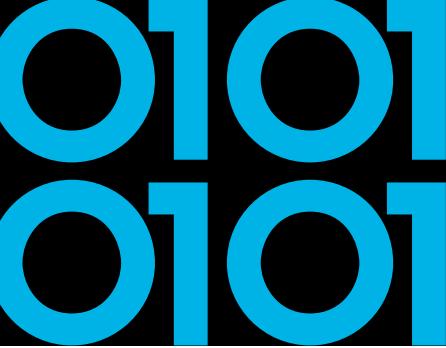
Dependency Confusion - NuGet

- Use single private repository
- Azure Artifacts can help manage and control upstream
- If your company has public packages consider registering prefix
- Lock packages with config



3rd Party Libraries

- Intent of library, know what's inside!
- Keep in mind that's a transitive list of dependencies
- Look out for talk on 'Sandboxing .NET Assemblies' I gave two weeks ago at NDC Porto!
- Open Source Security Foundation - OpenSSF
- Security Scorecards - Security health metrics for Open Source



Security Scorecards - OpenSSF

The screenshot shows a GitHub README page for the repository `ossf/scorecard`. The title is **Security Scorecards**. Below it, two green status indicators show `build passing` and `CodeQL passing`. The **Overview** section contains two bullet points:

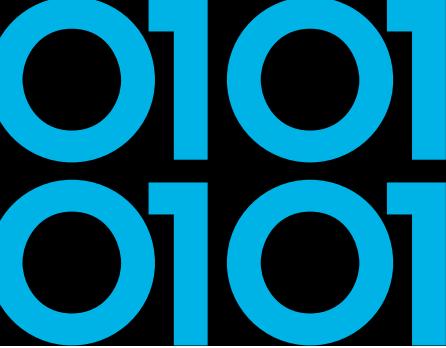
- What Is Scorecards?
- Prominent Scorecards Users

 The **Using Scorecards** section contains eight bullet points:

- Prerequisites
- Installation
- Authentication
- Basic Usage
- Report Problems
- Scorecards' Public Data

 At the bottom, there's a **Checks** section. To the right of the text, there's a cartoon illustration of a blue bird-like character wearing red shorts and a yellow shield, holding up a yellow sign with the number **10**.

Deps.Dev by Google



The screenshot shows a dark-themed web browser window with the title bar "Open Source Insights" and the URL "https://deps.dev". A green header bar at the top of the page reads "Open Source Insights is an experimental project by Google.". Below this, the main content area has a dark grey background with a light grey diagonal striped pattern. On the left, the text "open/source/insights" is displayed in white. On the right, there are navigation links for "About", "Documentation", and "Blog".

Understand your dependencies

Your software and your users rely not only on the code you write, but also on the code your code depends on, the code *that* code depends on, and so on. An accurate view of the complete dependency graph is critical to understanding the state of your project. And it's not just code: you need to know about security vulnerabilities, licenses, recent releases, and more.

Search for open source packages

All systems ▾

Search

/ 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.

0101
0101

Source Generators

A screenshot of a web browser window showing a blog post. The title of the post is ".NET 5, Source Generators, and Supply Chain Attacks". It is written by Mateusz Krzeszowiec and published on September 30, 2021. The post discusses IDEs and build infrastructure as targets for threat actors, mentioning XcodeGhost as an example. Below the post, there is a section for sharing the article on social media.

.NET 5, Source Generators, and Supply Chain Attacks

By **Mateusz Krzeszowiec** | September 30, 2021

Secure Development

Share this article: [Twitter](#) [Facebook](#) [LinkedIn](#)

IDEs and build infrastructure are being a target of various threat actors since at least 2015 when XcodeGhost has been discovered

- <https://en.wikipedia.org/wiki/XcodeGhost> - malware-ridden Apple Xcode IDE that enabled attackers to plant malware in iOS applications built using it.



Source Generators

- Any 3rd party library can contain Source Generator!
- Consider disabling on project-level:

```
<Target Name="DisableAnalyzers"
        BeforeTargets="CoreCompile">
    <ItemGroup>
        <Analyzer Remove="@(<Analyzer>)" />
    </ItemGroup>
</Target>
```



Reproducible/Deterministic Builds

The screenshot shows a website with a dark blue header featuring the Reproducible Builds logo. The main content area has a light gray background. On the left, there's a sidebar with a dark blue header containing the Reproducible Builds logo and the word "Reproducible". Below this, the sidebar lists several menu items: Home, Contribute, Documentation (which is highlighted in blue), Tools, Who is involved?, News, Events, and Talks. The main content area starts with a large heading "Definitions" in bold black font. Below it is another heading "When is a build reproducible?" in bold black font. To the right of this heading is a block of text explaining the definition of a reproducible build. At the bottom of the main content area is another block of text providing more context about the relevant attributes of the build environment.

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.

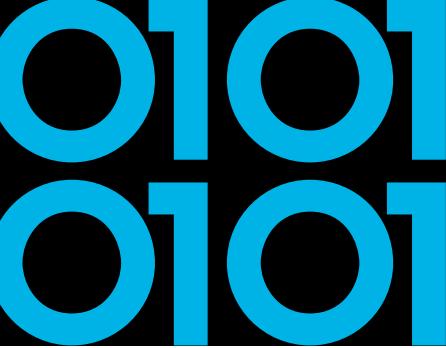


Reproducible/Deterministic Builds

- Roslyn v1.1 started supporting some kind of determinism on how items are emitted
- Given same inputs, the compiled output will always be deterministic
- Inputs can be found in Roslyn compiler docs
‘Deterministic Inputs’

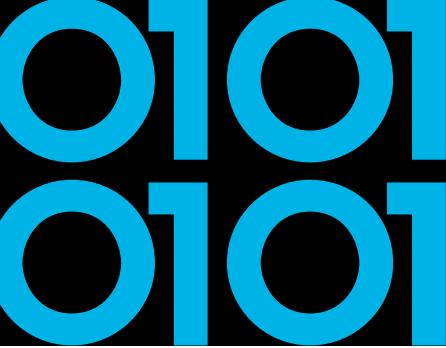
Reproducible Build .NET6

0101
0101



Reproducible/Deterministic Builds

- DotNet.Reproducible NuGet Package
 - MSBuild *ContinuousIntegrationBuild*
 - SourceLink
- Dotnet.Reproducible.Isolated NuGet Package
 - Hermetic builds



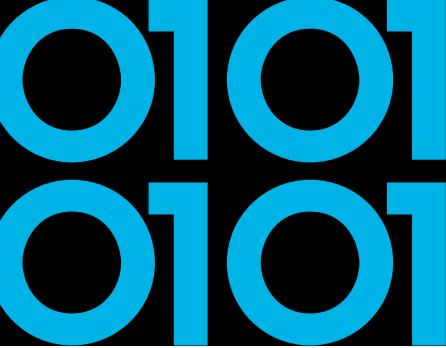
Reproducible Build Validation

- Design to validate NuGet packages & .NET binaries
 - Does linked source code match binaries?
 - Ability to rebuild reproducible based on given inputs
 - .NET CLI Validate tool `dotnet validate`

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, Docs, and a user icon. Below the navigation, the text "A new standard for signing, verifying and protecting software" is prominently displayed in large, bold, dark gray font. Underneath this, a smaller text "Making sure your software's what it claims to be." is visible. At the bottom, the text "In collaboration with" is followed by logos for ChainGuard, Cisco, Google, Hewlett Packard Enterprise, The Linux Foundation, Purdue University, Red Hat, and VMware.



Signing artifacts

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 --> SignatureLog["SIGNATURE TRANSPARENCY LOG"]; Monitor --> KeyLog["KEY TRANSPARENCY LOG"]; Fulcio --- TRUST_ROOT["TRUST ROOT"]; SignatureLog --- TRUST_ROOT; KeyLog --- TRUST_ROOT;
```

The diagram illustrates the sigstore workflow. At the top, a box labeled "DEVELOPERS, MAINTAINERS, MONITORS" contains three circular nodes: "SIGN AND PUBLISH ARTIFACTS", "PUBLISH SIGNING CERTIFICATES", and "MONITOR LOGS". Dotted lines connect each of these nodes to three corresponding rectangular boxes below: "FULCIO CERTIFICATE AUTHORITY", "SIGNATURE TRANSPARENCY LOG", and "KEY TRANSPARENCY LOG". Finally, dotted lines connect each of these three boxes to a large orange box at the bottom labeled "TRUST ROOT".

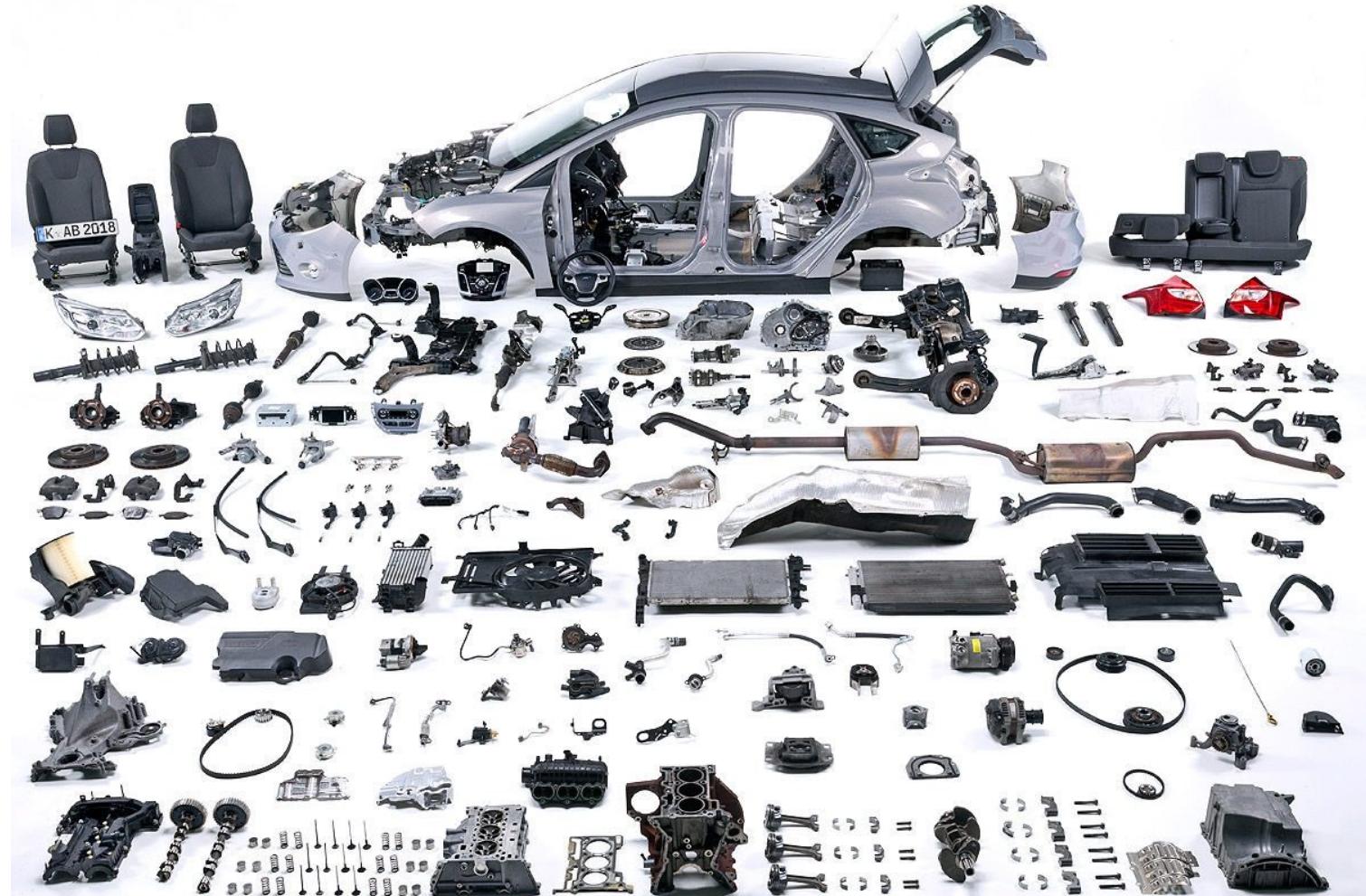


Signing artifacts

- 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



0101
0101

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

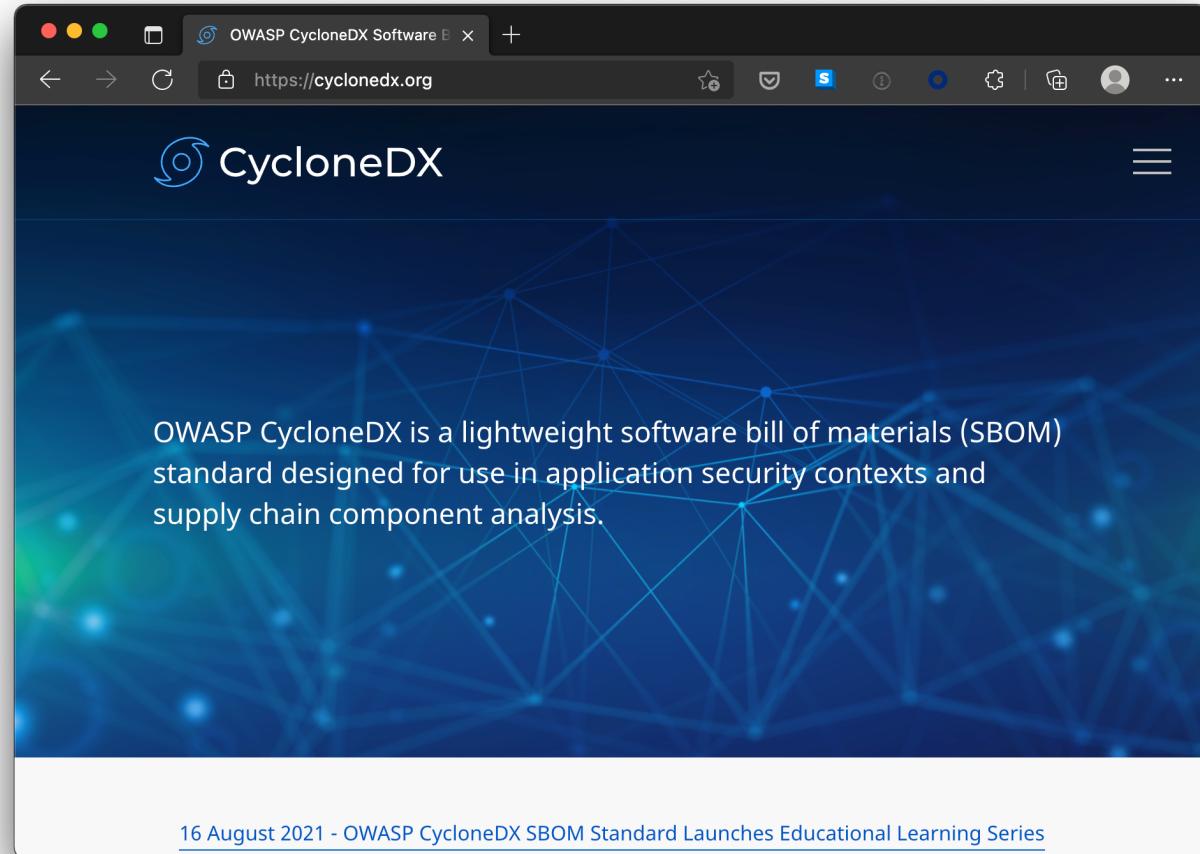


Software Bill of Materials (SBOM)

- 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? Process meets requirements?
 - Materials - How was the product created? Materials/Source used?

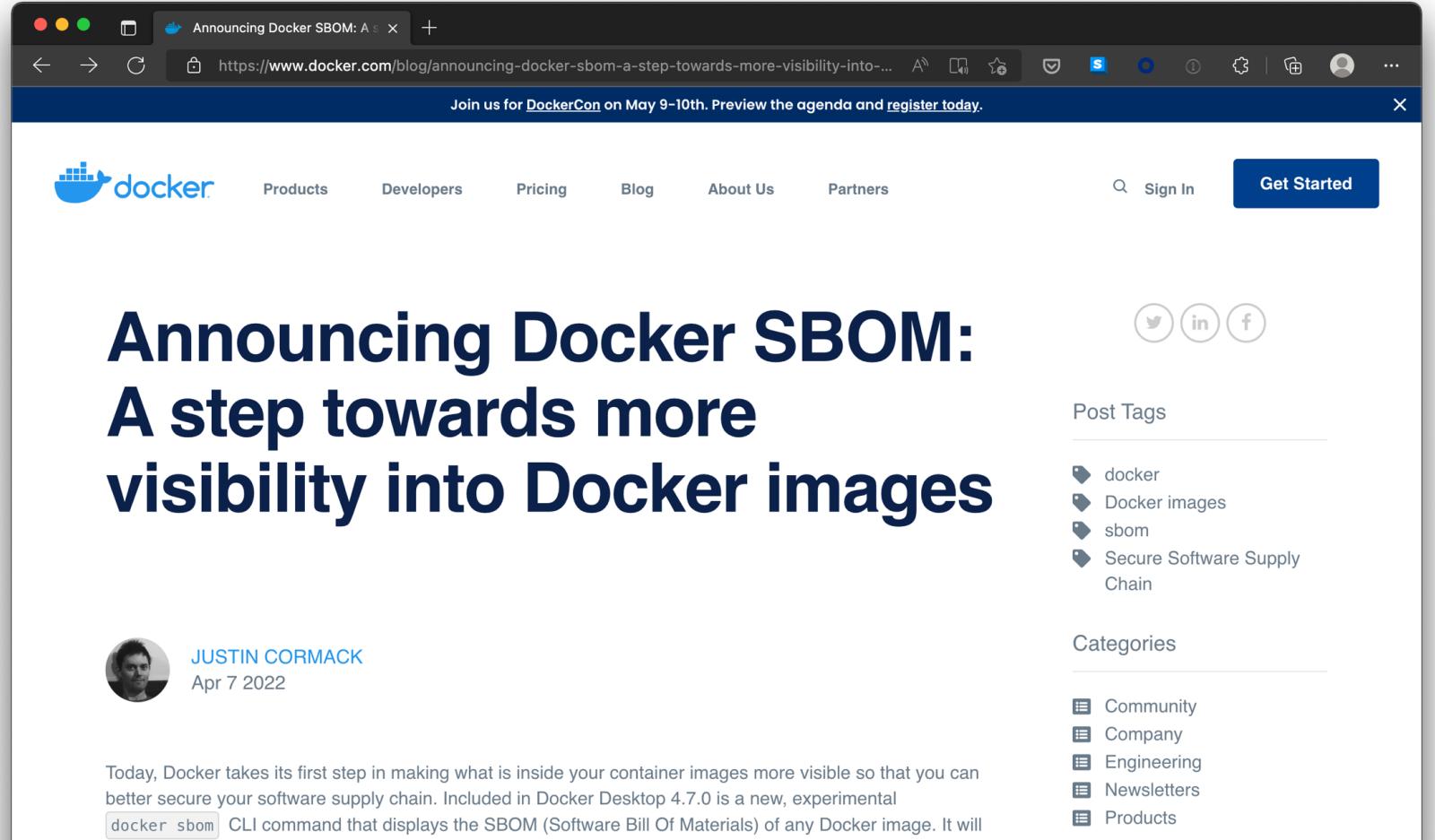
0101
0101

Software Bill of Materials (SBOM)



Docker SBOM

0101
0101



A screenshot of 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 post is authored by Justin Cormack and published on April 7, 2022. The Docker logo is visible in the top left corner of the page. The page includes a navigation bar with links to Products, Developers, Pricing, Blog, About Us, and Partners. There is also a "Get Started" button. On the right side, 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](#).

[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

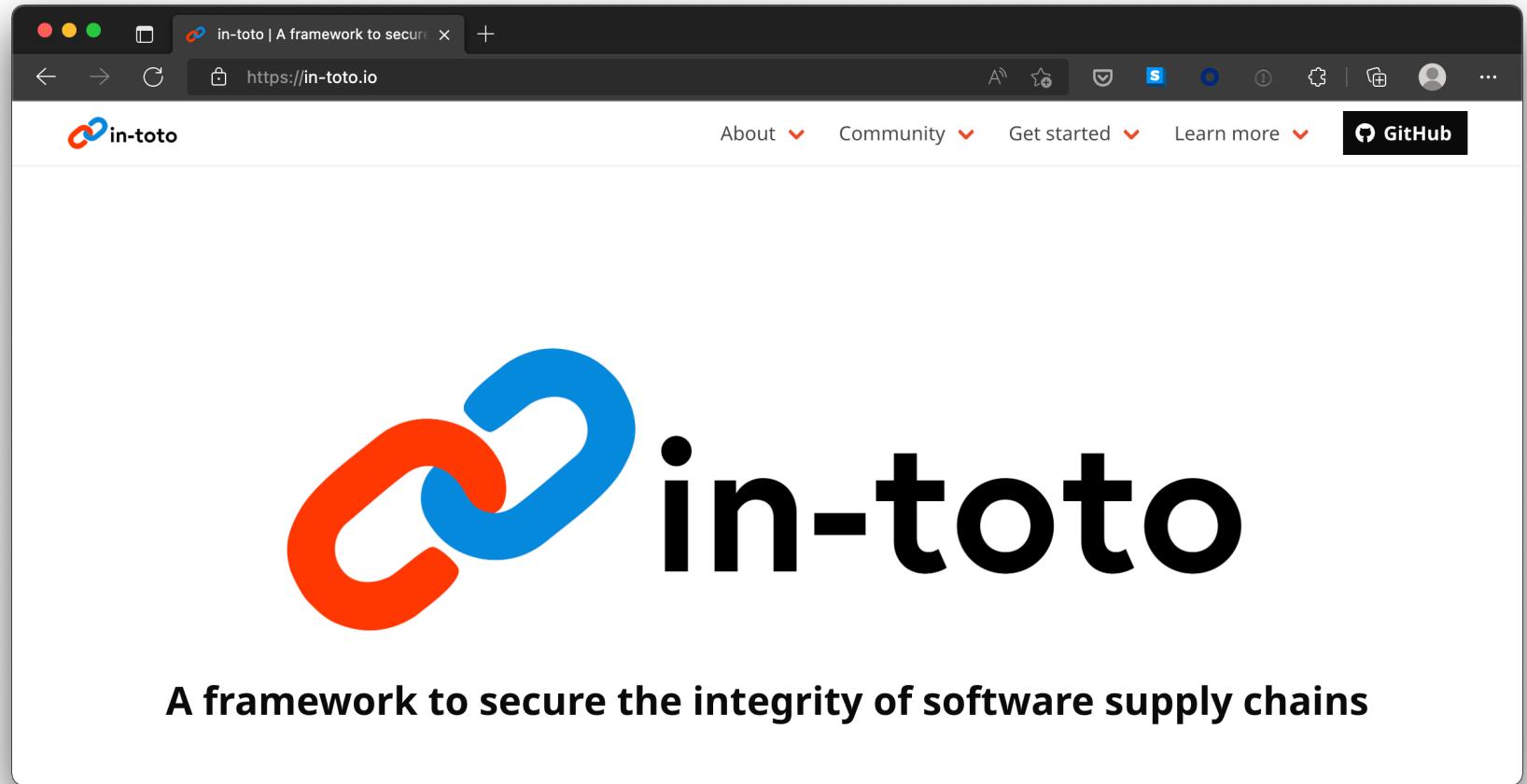
- # docker
- # Docker images
- # sbom
- # Secure Software Supply Chain

Categories

- Community
- Company
- Engineering
- Newsletters
- Products

In-toto

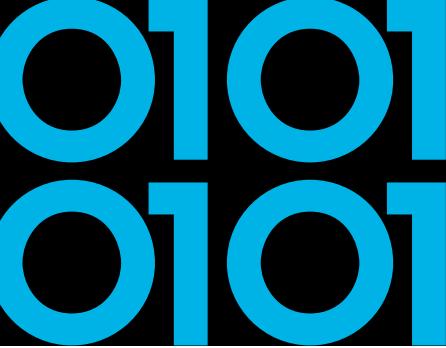
0101
0101





In-Toto - Terminology

- **Functionaries** that are identified by public key our supply chain.
For example, the Project-Owner, Developer, and Release Manager
- **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



In-Toto - Demo



In-Toto - Demo - Terminology

- **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

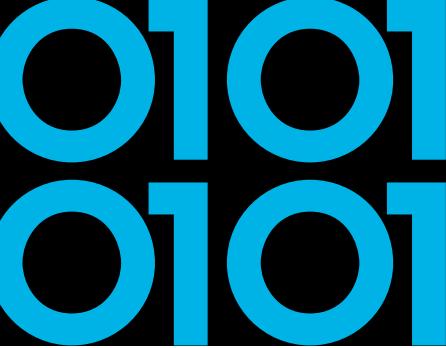
+17

0101
0101

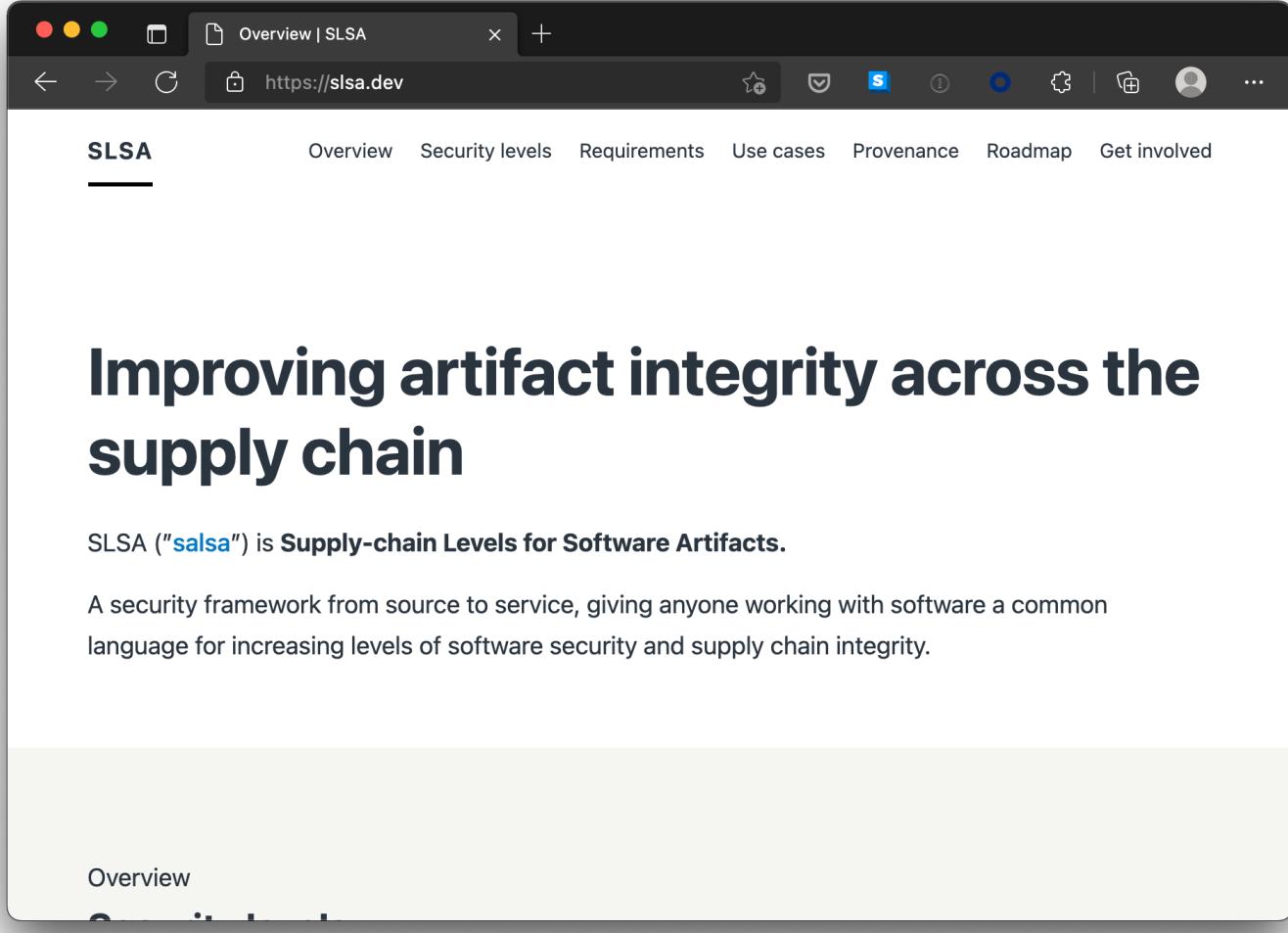
MyAwesomeWebApp Demo

- CycloneDX
- In-Toto
- Sigstore
- Docker SBOM





Google SLSA



The screenshot shows a web browser window displaying the SLSA website at <https://slsa.dev>. The page title is "Overview | SLSA". The navigation bar includes links for SLSA, Overview, Security levels, Requirements, Use cases, Provenance, Roadmap, and Get involved. The main content features a large heading: "Improving artifact integrity across the supply chain". Below the heading, a definition of SLSA is provided: "SLSA ("salsa") is Supply-chain Levels for Software Artifacts." A descriptive paragraph follows: "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 are links for "Overview", "Security levels", and "Requirements".

Improving artifact integrity across the supply chain

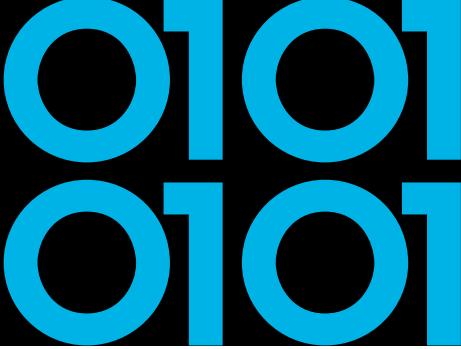
SLSA ("salsa") is **Supply-chain Levels for Software Artifacts**.

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

Security levels

Requirements



Google SLSA Levels

1. The build process must be fully scripted/automated and generate provenance.
2. Requires using version control and a hosted build service that generates authenticated provenance.
3. The source and build platforms meet specific standards to guarantee the auditability of the source and the integrity of the provenance respectively.
4. Requires two-person review of all changes and a hermetic, reproducible build process.



SLSA GitHub Action

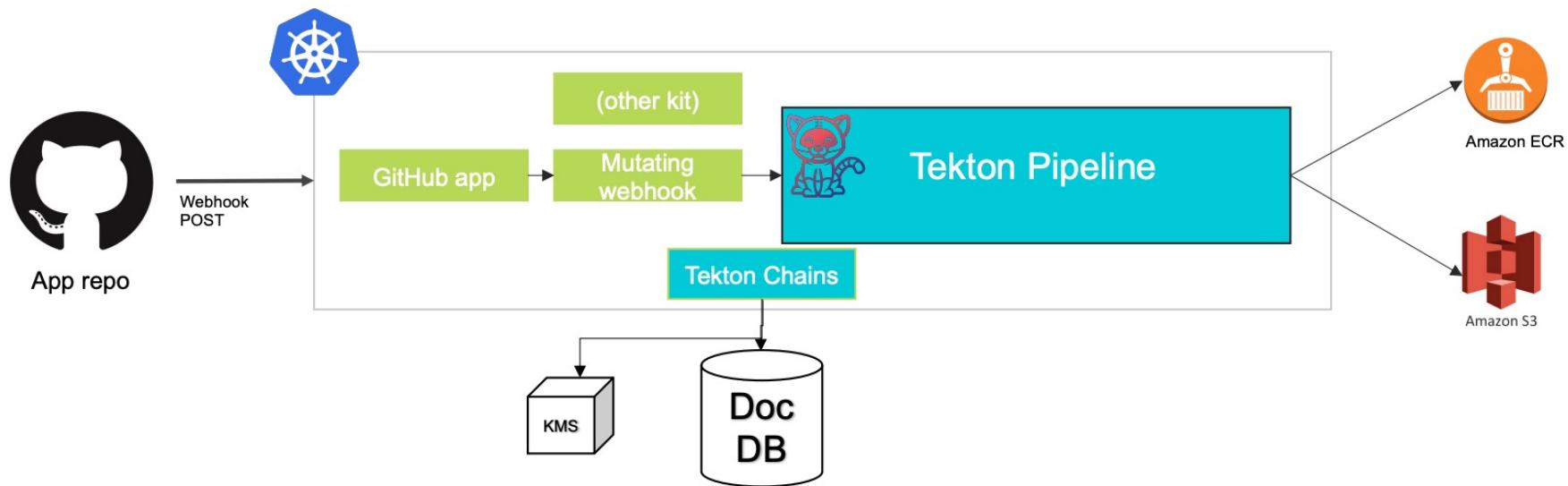
- Released April 2022
- SLSA level 2 provenance generator in GitHub Action
- SLSA level 3+ provenance generator for Go binaries
- GitHub Hosted Runner
- Uses SigStore to do keyless signing with GitHub OIDC
- Verifier included



SolarWinds Project Trebuchet



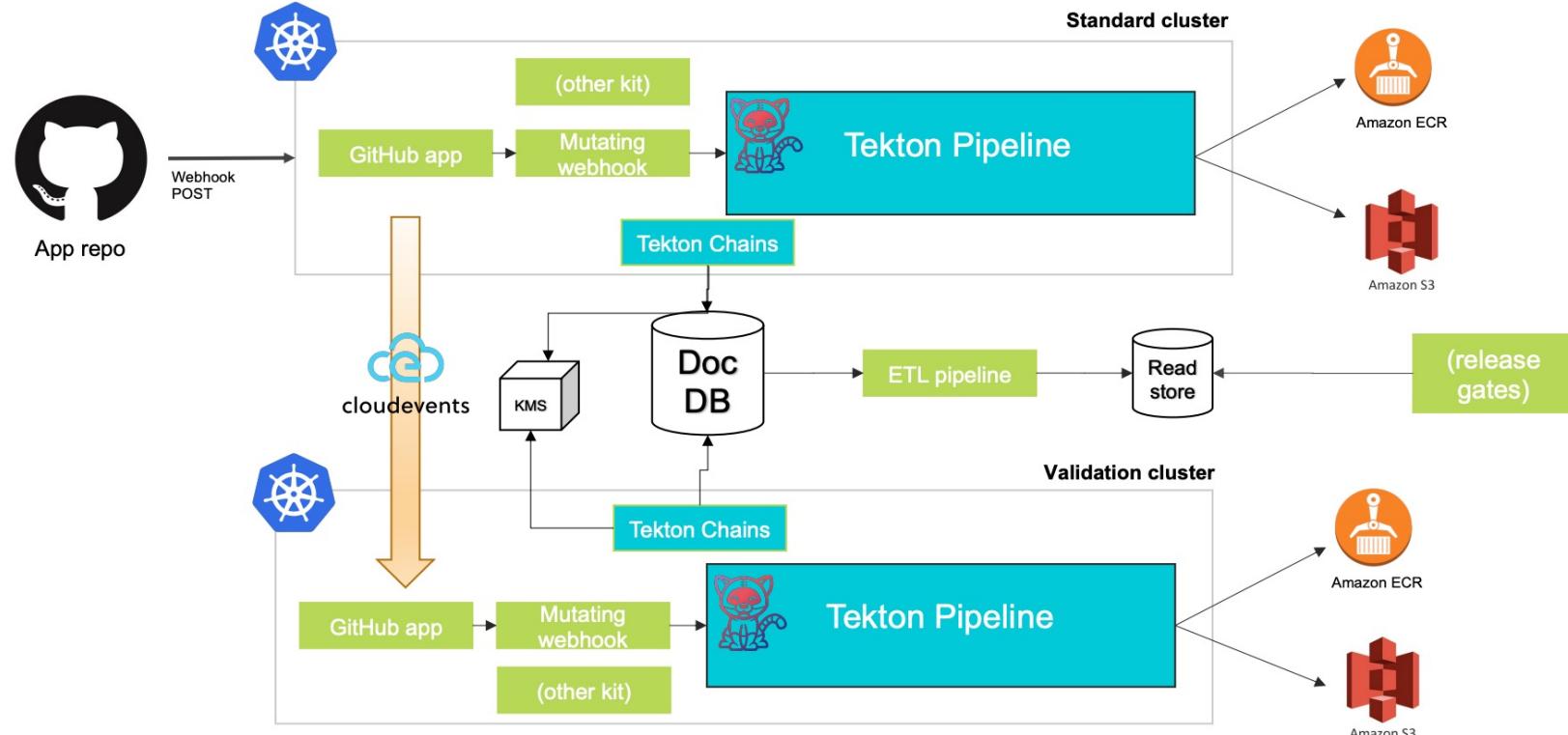
Pipeline With Attestations



SolarWinds Project Trebuchet



Reading Results



IBM OpenShift

0101
0101

A screenshot of a web browser displaying a Medium article. The article title is "A Zero Trust Approach for Securing the Supply Chain of Microservices Packaged as Container Images" by Gerry Kovan. The article summary states: "Securing the software supply chain has become a top priority for both open source as well as closed source projects. There are many sources in a software supply chain such operating systems and operating system packages/binaries, programming language frameworks and libraries, shell". The Medium interface shows 28 followers, a "Follow" button, and social sharing icons for Twitter, Facebook, LinkedIn, and others.

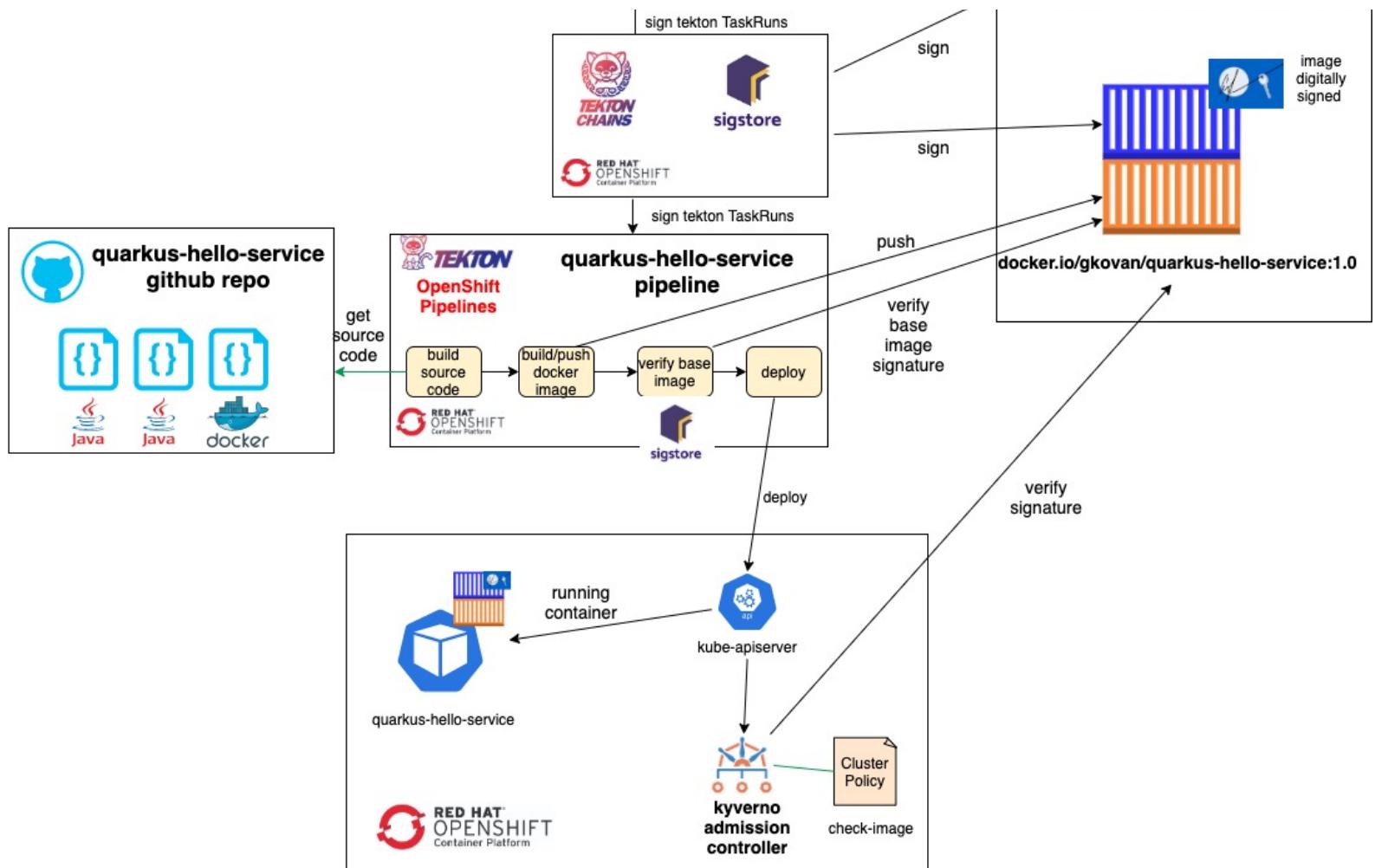
A Zero Trust Approach for Securing the Supply Chain of Microservices Packaged as Container Images

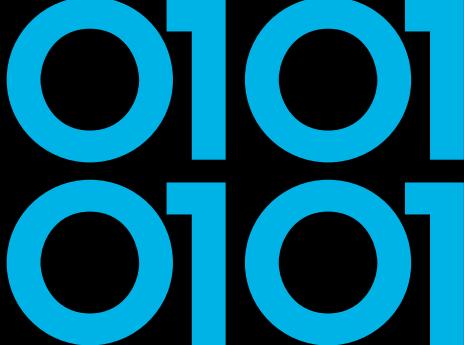
Gerry Kovan · Sep 6 · 6 min read

Securing the software supply chain has become a top priority for both open source as well as closed source projects. There are many sources in a software supply chain such operating systems and operating system packages/binaries, programming language frameworks and libraries, shell



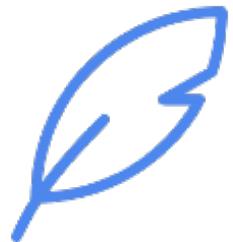
IBM OpenShift





Grafeas and Kritis by Google

- Grafeas - Component Metadata API
 - Container Analysis API on Google Cloud Platform
- Kritis - Deployment Authorization for Kubernetes Apps
 - Binary Authorization on Google Cloud Platform



Azure Pipelines Artifact Policy

0101
0101

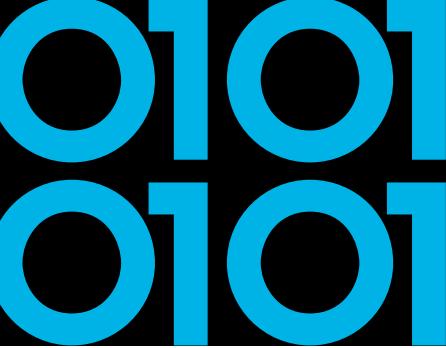
The screenshot shows a web browser window displaying the Microsoft Azure DevOps Documentation. The title bar reads "Artifact policy checks - Azure". The URL in the address bar is <https://docs.microsoft.com/en-us/azure/devops/pipelines/process/artifact-pol...>. The page content is titled "Artifact policy checks" and discusses artifact policies for deployment to critical environments like production. It includes sections on Prerequisites, Creating custom policies, and a note about evaluating artifacts against deployable artifacts in a pipeline run. The left sidebar shows navigation options for Azure DevOps Services, including "Configure security & settings", "Migrate", "Pipeline tasks", "Troubleshooting", and "Reference". The Microsoft logo is at the top left, and there are "Docs", "Documentation", "Learn", "Q&A", and "Code Samples" links. A "Try for free" button is visible on the right.

Chainguard Enforce

0101
0101

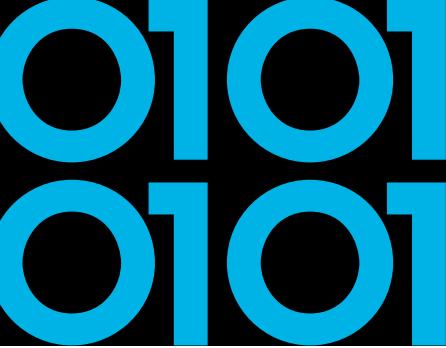
The screenshot shows a web browser window for 'The New Stack' website. The URL is <https://thenewstack.io/chainguard-enforce-software-supply-chain-security-for-k8s>. The page title is 'Chainguard Enforce: Software Supply Chain Security for K8s'. Below the title is a black and white photograph of a padlock and chain. A small blue shield icon is in the bottom left corner.

The screenshot shows a web browser window for 'Chainguard Enforce' website. The URL is <https://www.chainguard.dev/chainguard-enforce-software-supply-chain-security-for-k8s>. The page features a large purple header with the Chainguard logo and navigation links for Solutions, Resources, and Company. The main heading is 'Chainguard Enforce'. To the right, there is a text block: 'Enforce enables you to build, manage, ensure continuous compliance, and enforce policies that protect your organization from supply chain threats.' Below this are two buttons: 'Request a demo' and 'Preview'.



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/ndclondon2022>

ntanis at veracode.com

@nielstanis on Twitter

