

NDC { Oslo }

Securing your .NET application
software supply-chain

Niels Tanis

VERACODE

0101
0101

Who am I?

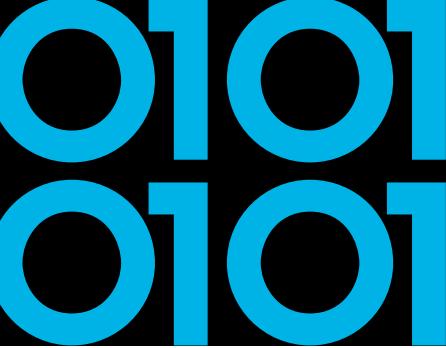
- Niels Tanis
 - Security Researcher @ Veracode
 - Background in .NET Development
 - Application Security Consultancy
 - Pen-testing & Ethical Hacking
 - ISC² CSSLP



Securing your .NET application software supply-chain

0101
0101





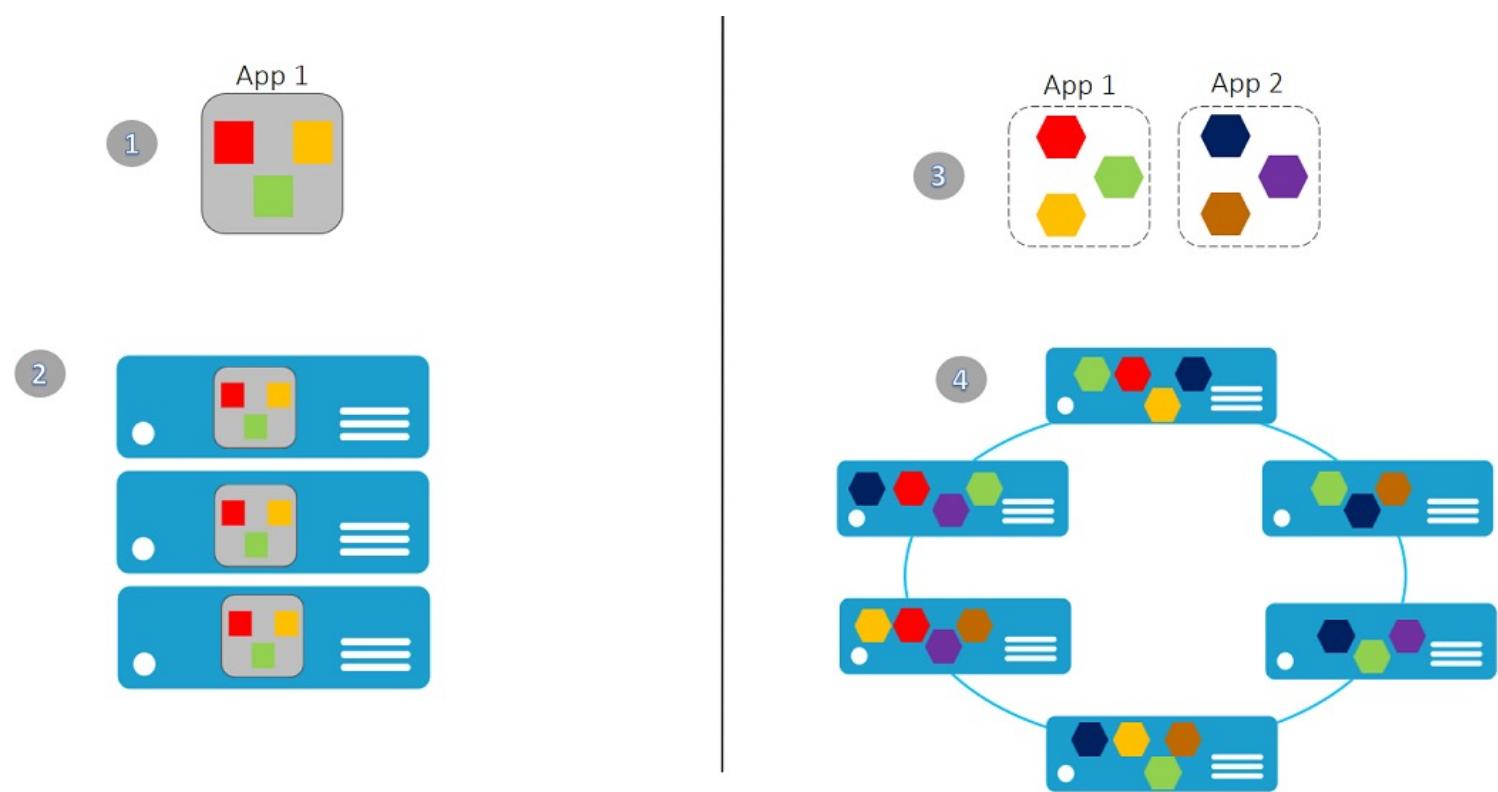
Agenda

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

Evolution in Software Architecture

0101
0101

- Monolith
- Microservices
- Serverless
- Cloud-Native



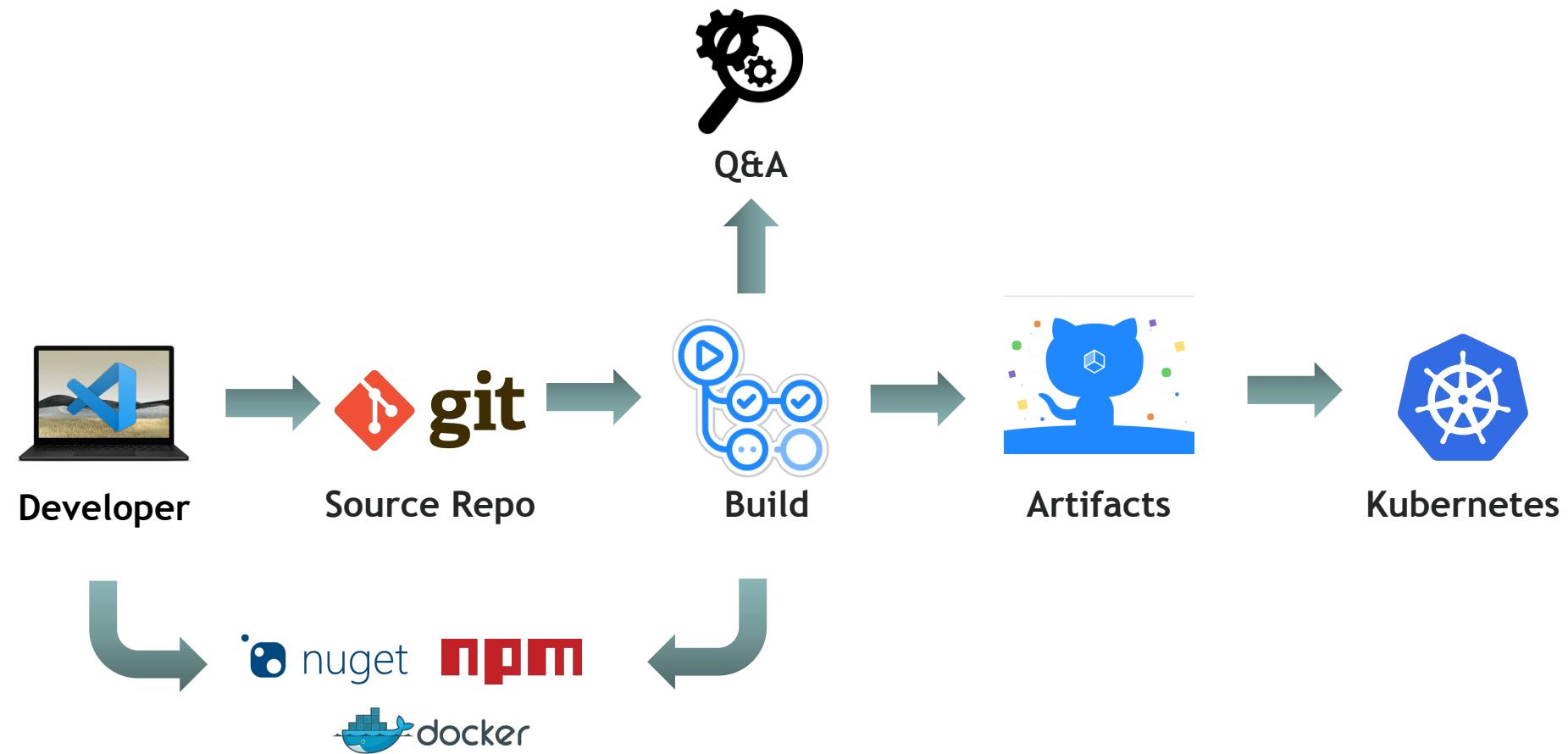
0101
0101

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 attributed to the CrowdStrike Intelligence Team under the category "Research & Threat Intel". The background of the page features a large, stylized graphic of a sun with rays emanating from it.

SUNSPOT: An Implant in the Build Process

January 11, 2021 CrowdStrike Intelligence Team Research & Threat Intel

The graphic consists of a central orange sphere with numerous thin, glowing red and yellow lines radiating outwards, resembling the sun's corona or a solar flare.



Kaseya

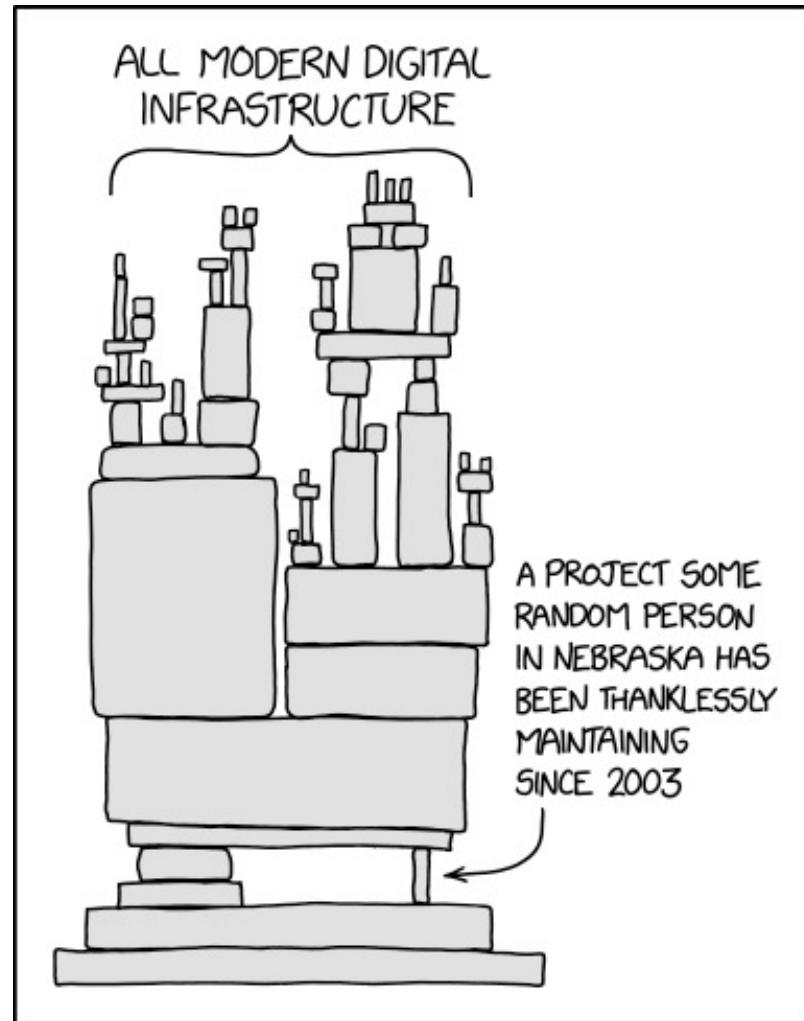
Kaseya Ransomware: a Software Supply Chain Attack or Not?

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



0101
0101

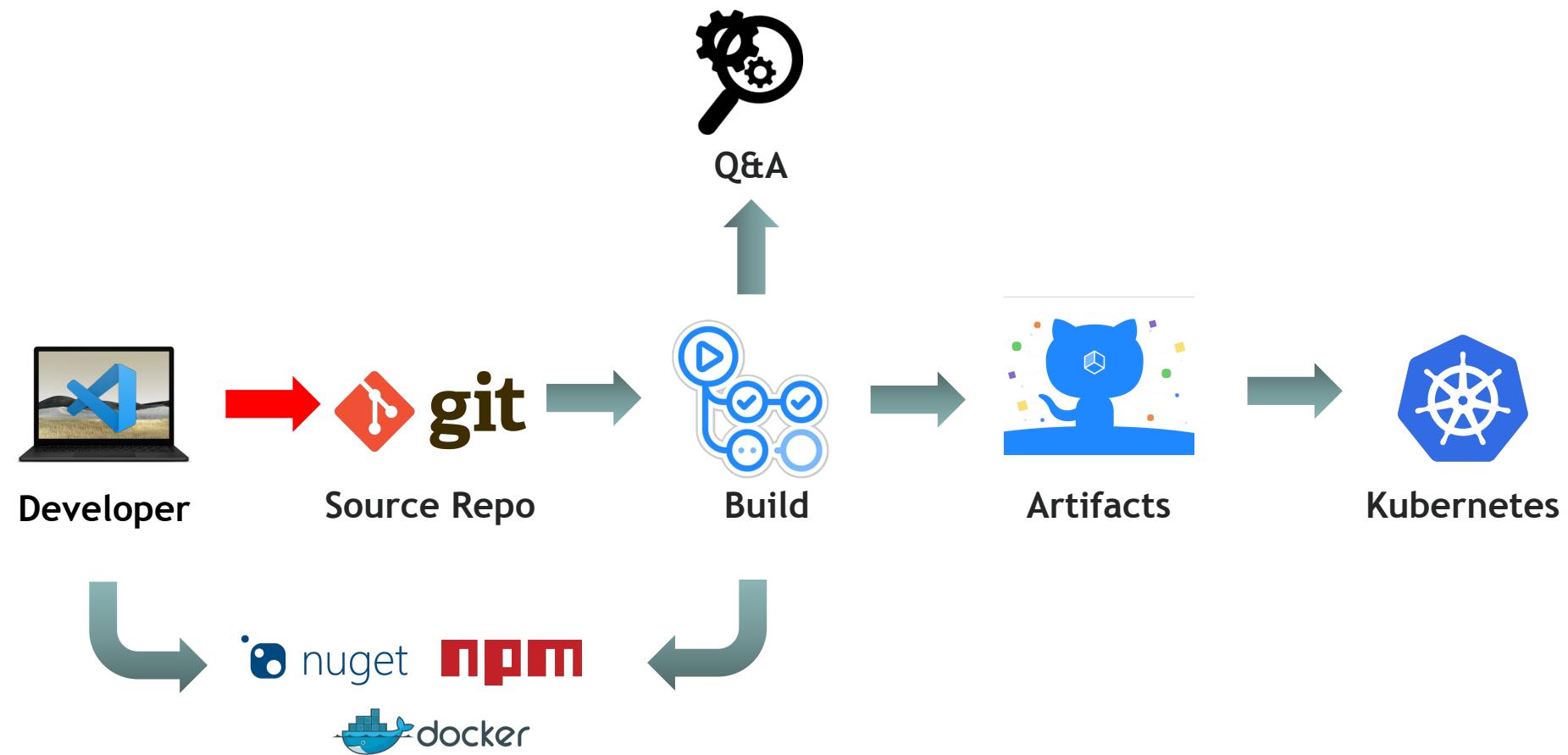
XKDC - Dependency



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

Software Supply Chain

0101
0101



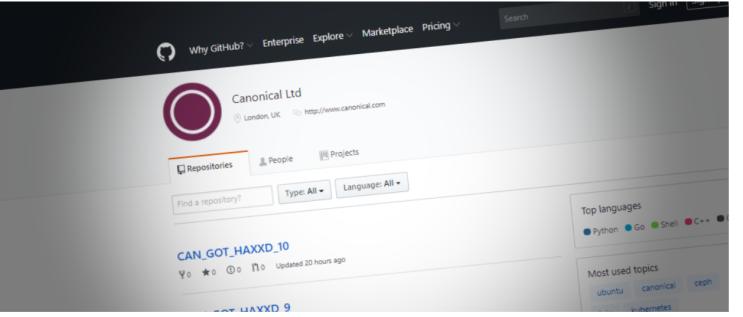
GitHub account

0101
0101

Canonical GitHub account hacked, Ubuntu source code safe

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

By Catalin Cimpanu for Zero Day | July 7, 2019 | Topic: Security



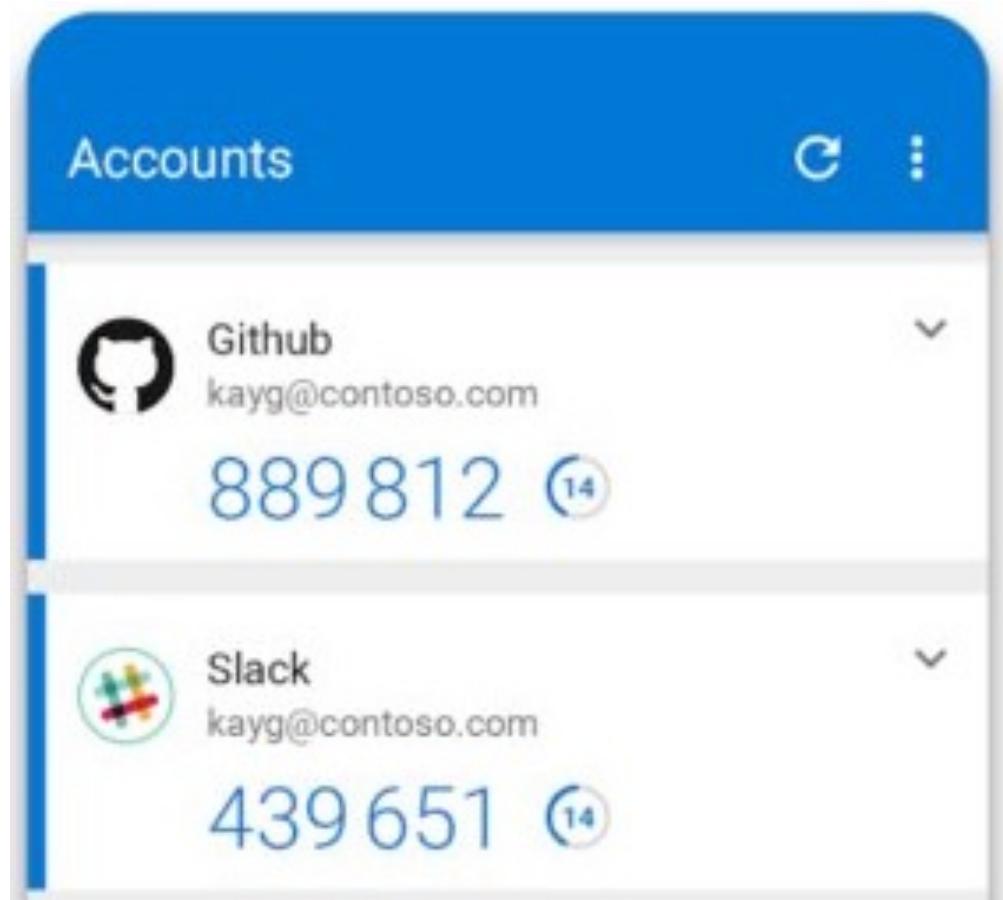
The GitHub account of Canonical Ltd., the company behind the Ubuntu Linux distribution, has been hacked. A file named CAN_GOT_HAXXED_10 was recently updated, which is a common tactic used by hackers to indicate they have gained access to a system. Canonical is investigating the incident.

RELATED

Why everyone should have this cheap security tool

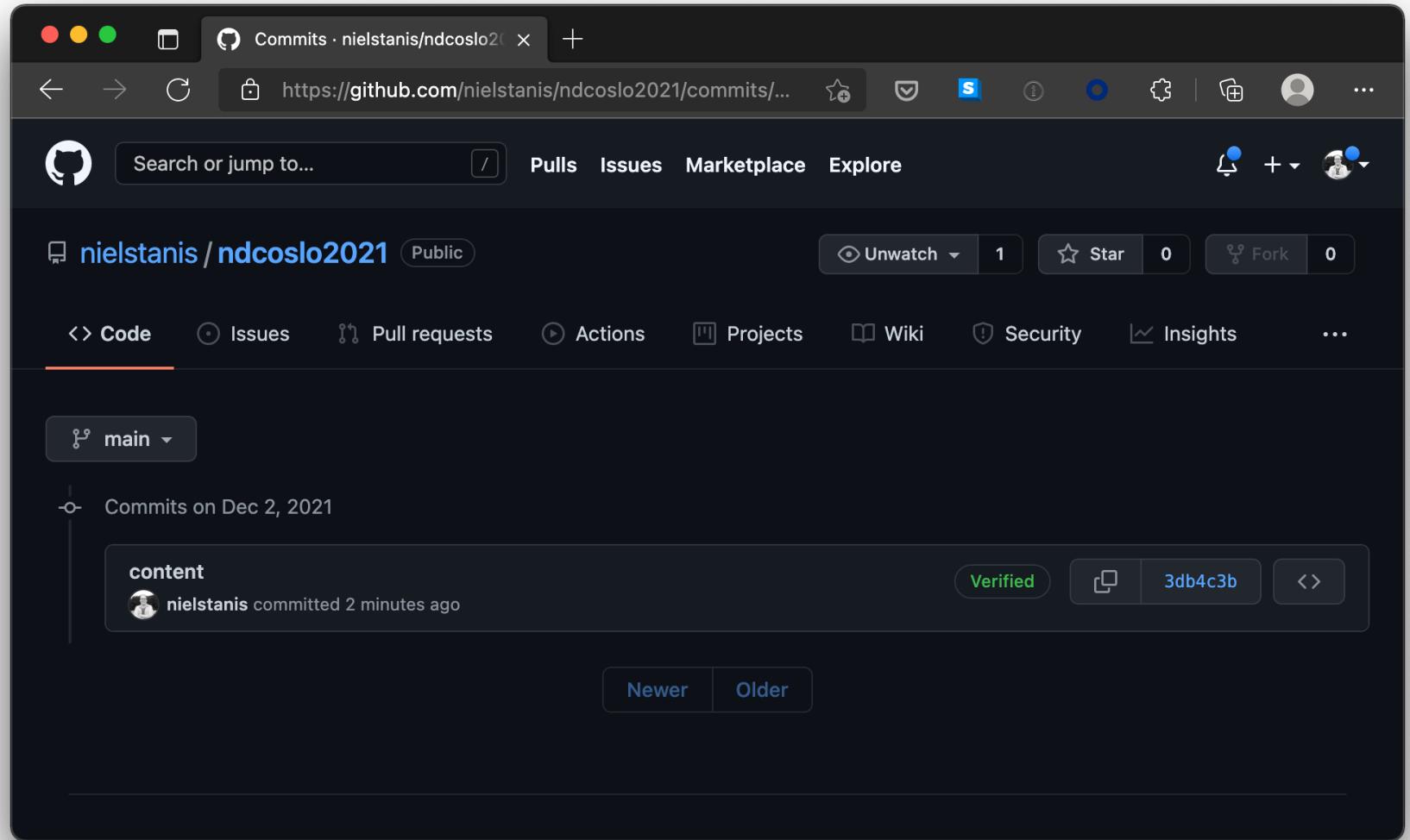
0101
0101

Use MFA on source-repository



GIT Commit Signing

0101
0101



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...>. 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

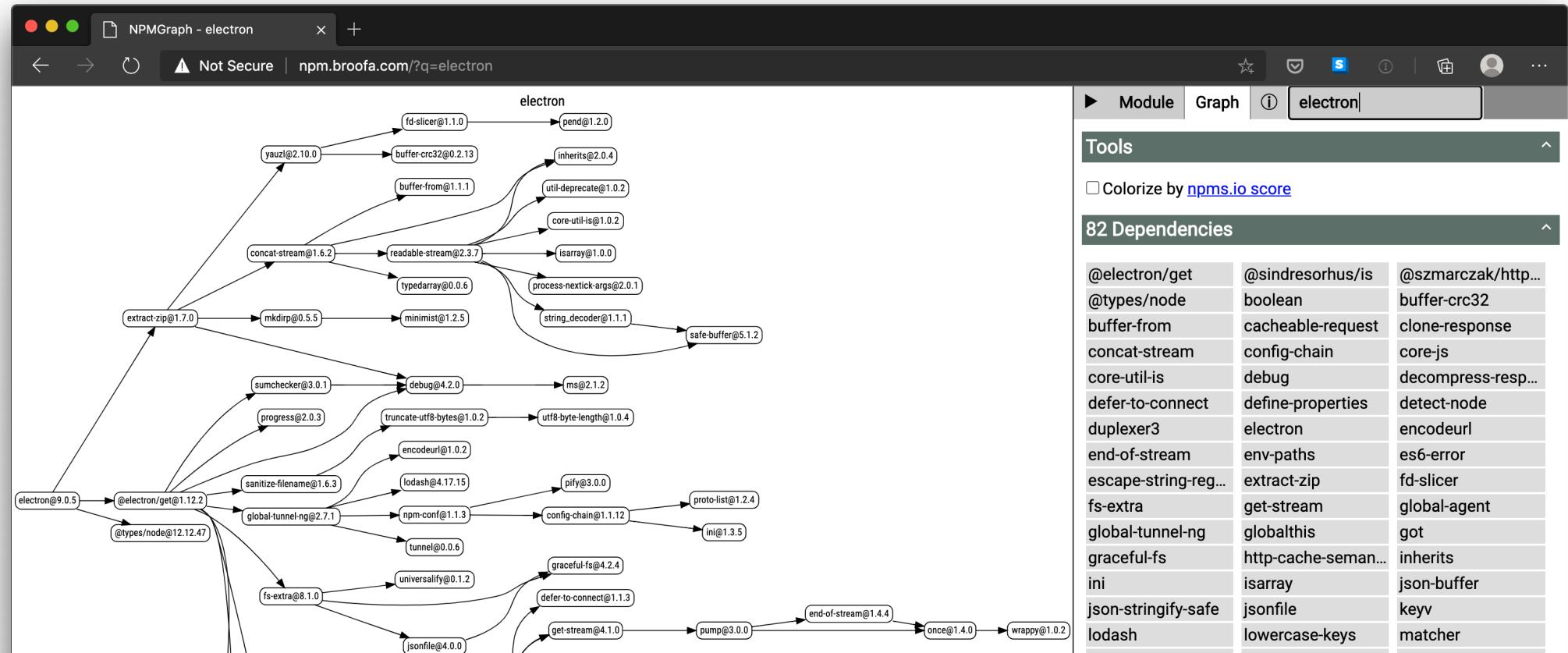


Alvaro Muñoz

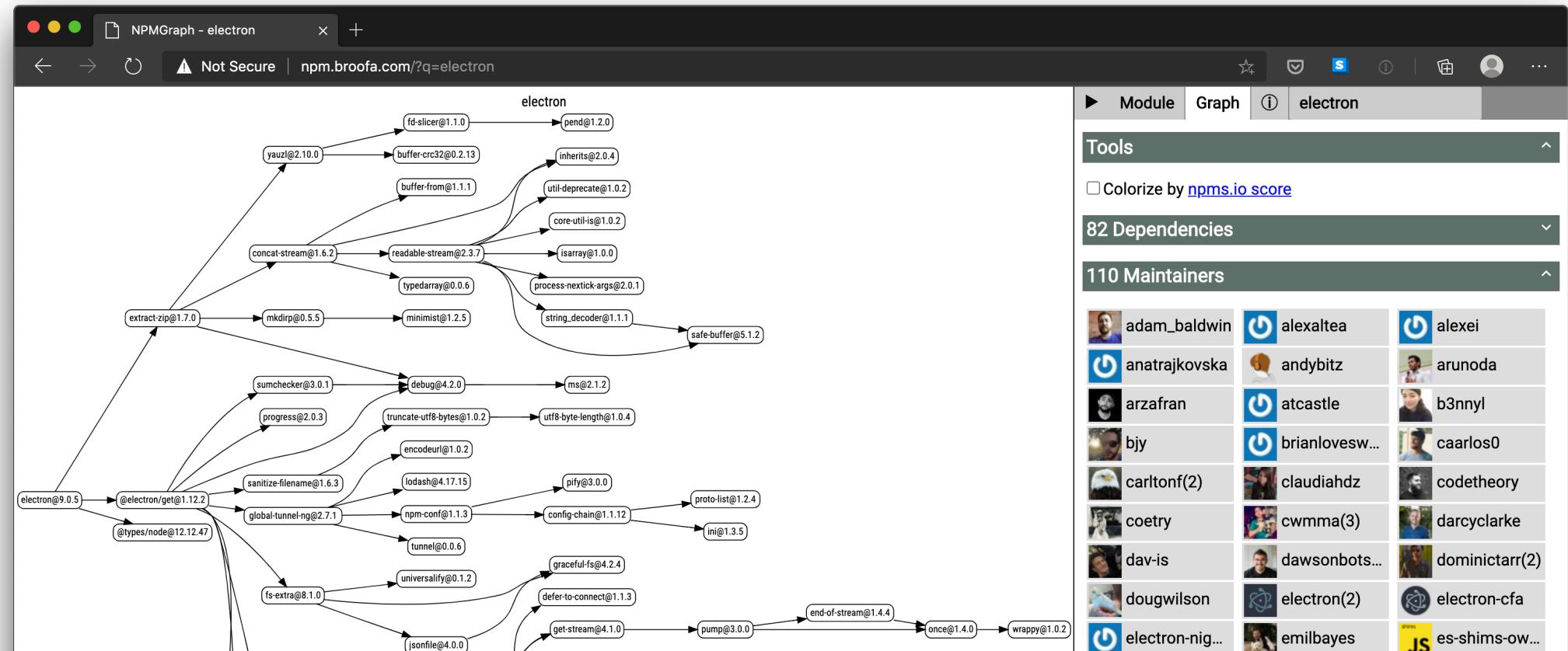
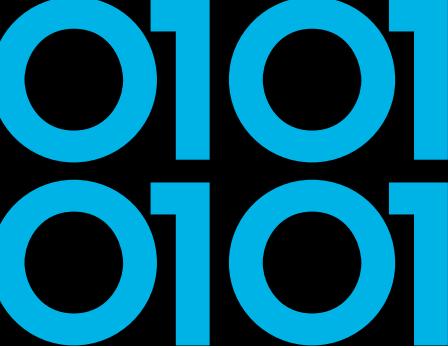
Securing the open source supply chain is an enormous task. It goes far beyond a security assessment or just patching for the latest CVEs. Supply chain security is about the integrity of the entire software development and delivery ecosystem. From the code commits themselves, to how they flow through 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

0101
0101



Visual Studio Code



Visual Studio Code

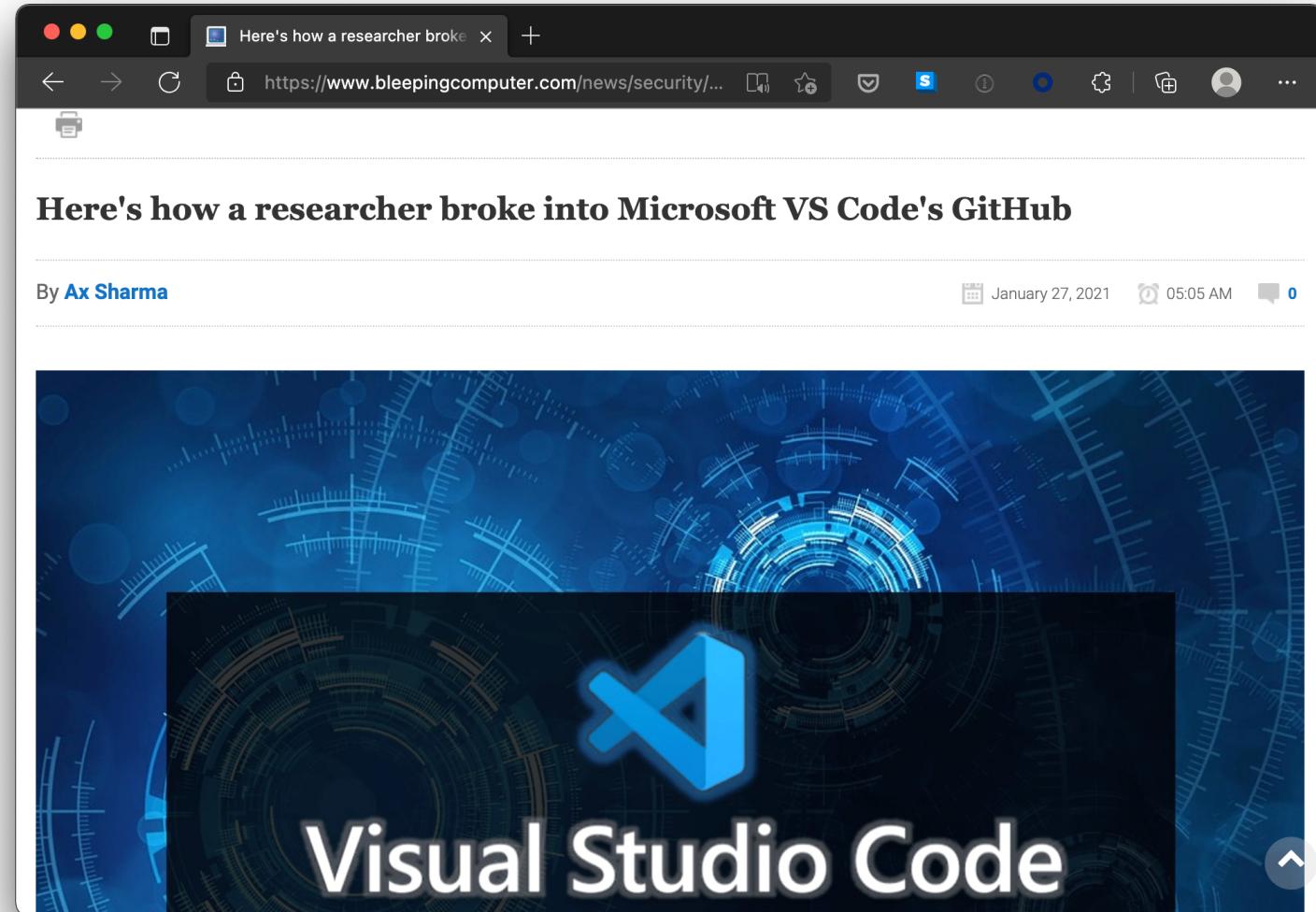
0101
0101

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

- Title Bar:** CVE-2021-42322 - Security Update
- URL:** https://msrc.microsoft.com/update-guide/vulnerabilities/CVE-2021-42322
- Header:** Microsoft | MSRC | Security Updates | Acknowledgements | Developer
- Breadcrumbs:** MSRC > Customer Guidance > Security Update Guide > Vulnerabilities > CVE 2021 42322
- 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 Elevation of Privilege Vulnerability
- CVE ID:** CVE-2021-42322
- On this page:** Security Vulnerability
- Released:** Nov 9, 2021
- Assigning CNA:** Microsoft
- MITRE CVE ID:** MITRE CVE-2021-42322
- CVSS Score:** CVSS:3.1 7.8 / 6.8

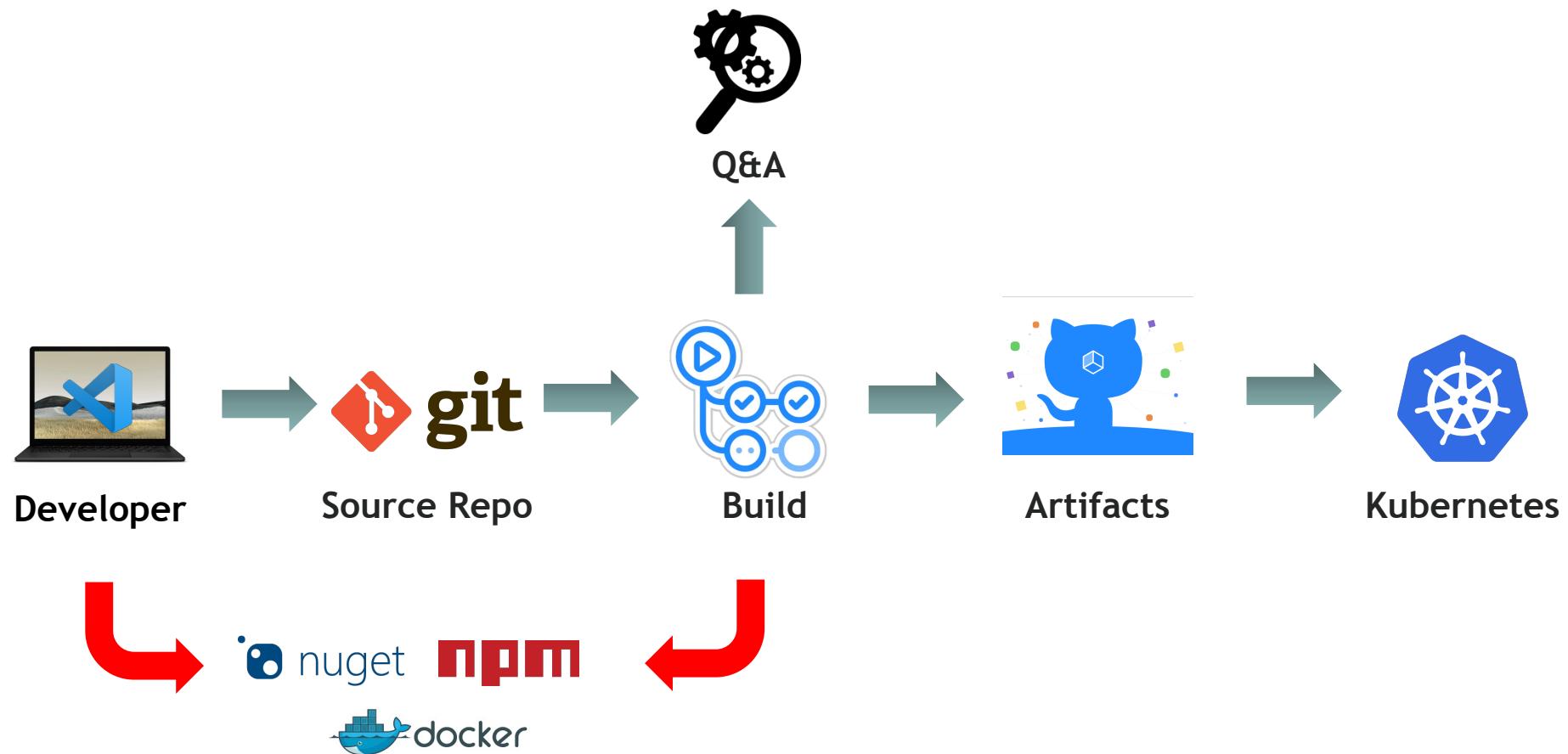
0101
0101

Visual Studio Code



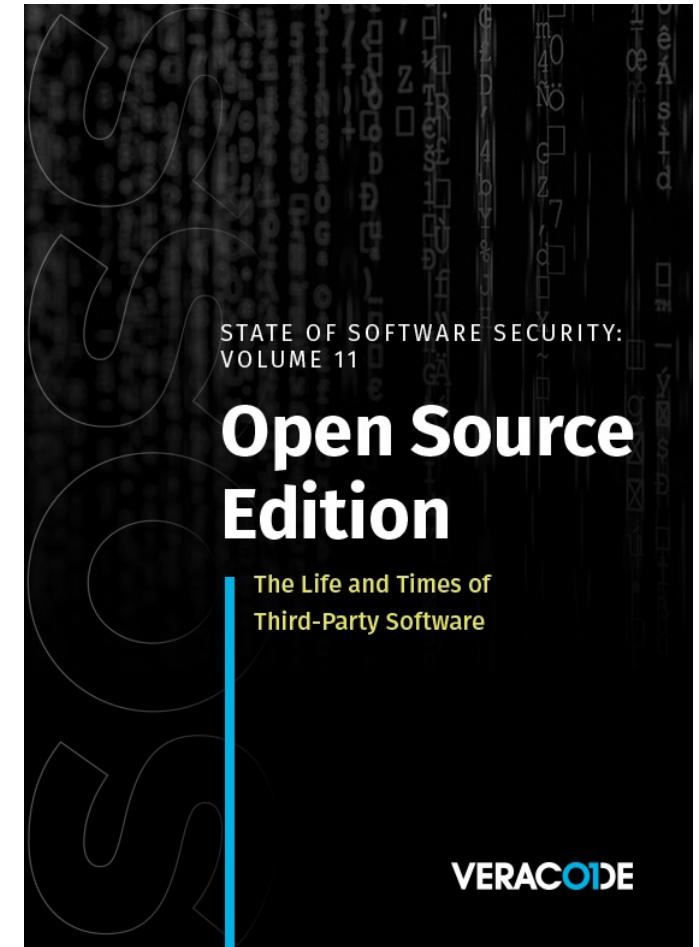
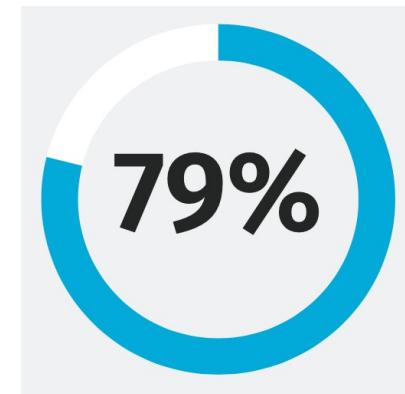
3rd Party Libraries

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.”*



Vulnerabilities in libraries

0101
0101

The screenshot shows a GitHub issue page for the repository `dotnet/announcements`. The issue is titled "Microsoft Security Advisory CVE-2021-41355 | .NET Core Information Disclosure Vulnerability #202". The issue was opened by `rjhanda` on 12 Oct and has 0 comments. The issue body contains a summary of the vulnerability, stating that an Information Disclosure vulnerability exists in .NET where `System.DirectoryServices.Protocols.LdapConnection` may send credentials in plain text on Linux and macOS. The issue is labeled with ".NET 5.0" and "Security". The GitHub interface includes a navigation bar with links for Code, Issues (176), Pull requests, Security, and Insights.

Microsoft Security Advisory CVE-2021-41355 | .NET Core Information Disclosure Vulnerability #202

Open rjhanda opened this issue on 12 Oct · 0 comments

rbhanda commented on 12 Oct · edited

Microsoft Security Advisory [CVE-2021-41355](#) | .NET Core Information Disclosure Vulnerability

Executive summary

Microsoft is releasing this security advisory to provide information about a vulnerability in .NET. This advisory also provides guidance on what developers can do to update their applications to remove this vulnerability.

An Information Disclosure vulnerability exists in .NET where `System.DirectoryServices.Protocols.LdapConnection` may send credentials in plain text on Linux and macOS.

Assignees
No one assigned

Labels
.NET 5.0 Security

Projects
None yet

Milestone
No milestone

Linked pull requests
Successfully merging a pull request may

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



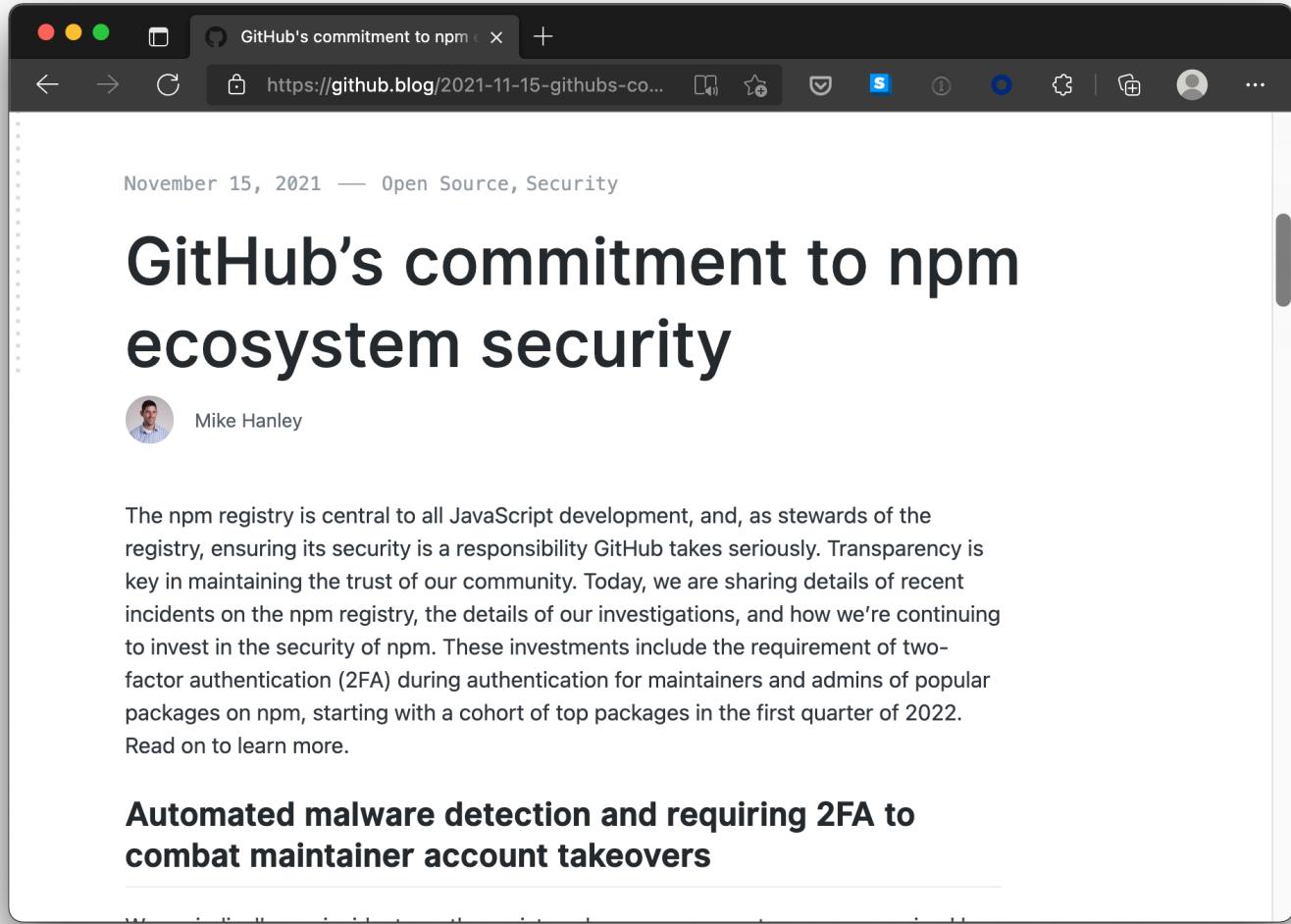
Versions of a popular NPM package named [ua-parser-js](#) was found to contain malicious code. [ua-parser-js](#) is used in apps and websites to discover the type of device or browser a person is using from User-Agent data. A computer or device with the affected software installed or running could allow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

0101
0101

Vulnerabilities in libraries



A screenshot of a web browser window showing a blog post from GitHub. The title of the post is "GitHub's commitment to npm ecosystem security". The author is Mike Hanley. The post discusses GitHub's efforts to ensure the security of the npm registry, including automated malware detection and requiring two-factor authentication (2FA) for maintainers and admins of popular packages.

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

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 has a subtitle "The Story of a Novel Supply Chain Attack". Below the author's name is a timestamp "Feb 9 · 11 min read". To the right of the author information are social sharing icons for Twitter, Facebook, LinkedIn, and a link icon. Below the article title is a large image of several colorful shipping containers stacked together against a clear blue sky.



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

WordPress Plugin Confusion



./ Kamil Vavra | @vavkamil

Offensive website security | Bug bounty | Ethical hacking

[Whoami](#) [Bug bounty](#) [Blog](#) [Tools](#) [Talks](#) [LinkedIn](#) [Contact](#)

25 November 2021

WordPress Plugin Confusion: How an update can get you pwned

by vavkamil

20 minutes to read

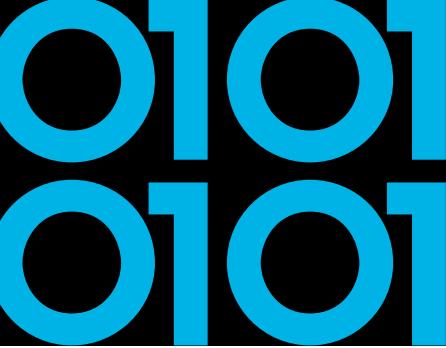
tl;dr: Like the novel “[Dependency Confusion](#)” supply chain attack, it is possible to take over internally developed WordPress plugins unclaimed on the wordpress.org registry. Updating the plugin might result in the RCE or installing a PHP backdoor. You can use [wp_update_confusion.py](#) to scan for potential targets. To protect your website, please read this [announcement](#).



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 in November.

- Open Source Security Foundation - OpenSSF
- Security Scorecards - Security health metrics for Open Source

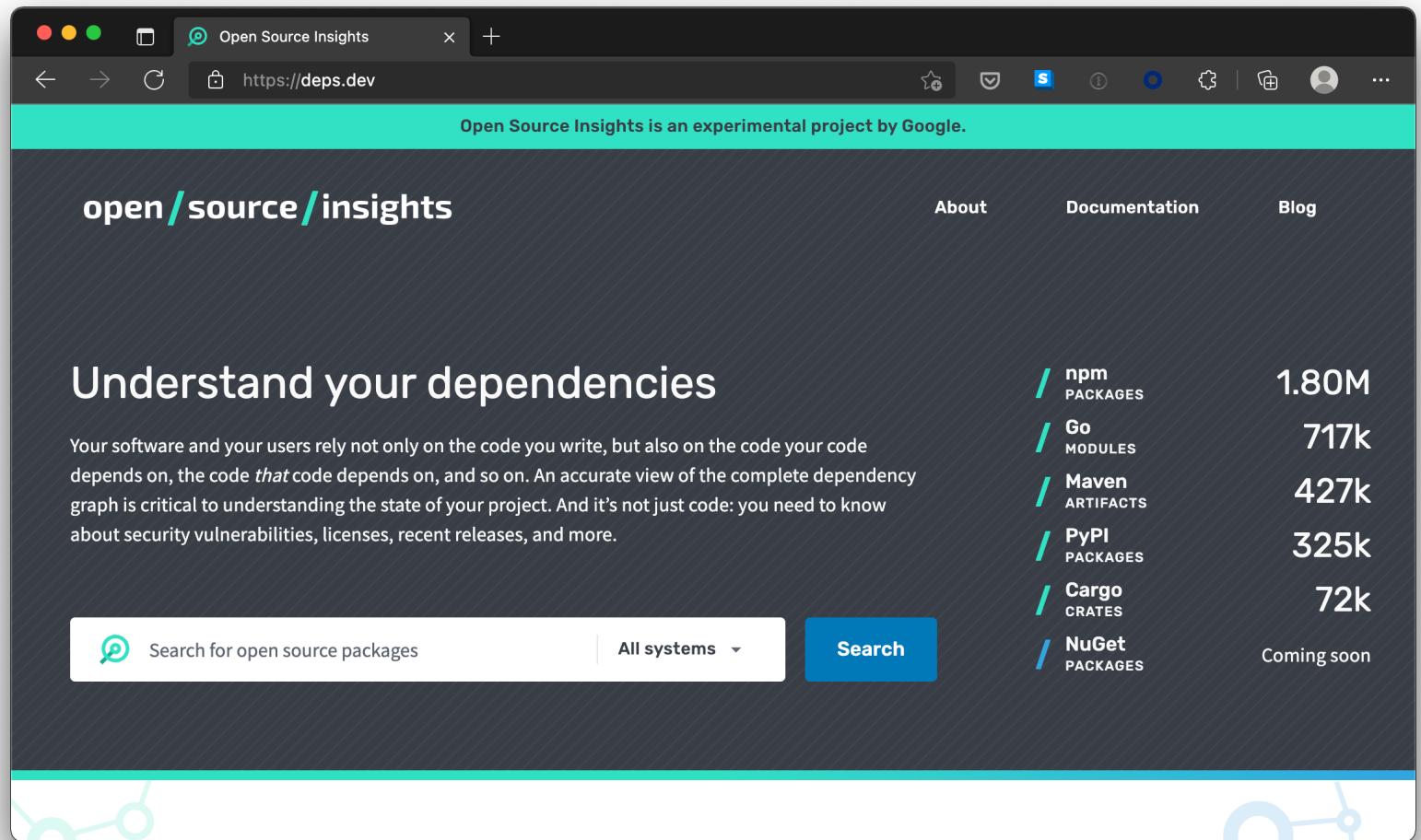


Security Scorecards - OpenSSF

A screenshot of a web browser displaying the GitHub README page for the "ossf/scorecard" repository. The page has a dark theme. At the top, there's a navigation bar with icons for back, forward, search, and user profile. Below it is a header with the repository name and a link to the URL https://github.com/ossf/scorecard. The main content area starts with a section titled "README.md" and "Security Scorecards". It includes status badges for "build" and "CodeQL" both labeled "passing". There are three main sections: "Overview", "Using Scorecards", and "Checks". The "Overview" section contains two bullet points: "What Is Scorecards?" and "Prominent Scorecards Users". The "Using Scorecards" section contains eight bullet points: "Prerequisites", "Installation", "Authentication", "Basic Usage", "Report Problems", and "Scorecards' Public Data". The "Checks" section is currently empty. To the right of the text, there is 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" on it. The entire screenshot is set against a white background.

Deps.Dev by Google

0101
0101



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 navigation bar includes 'open/source/insights', 'About', 'Documentation', and 'Blog'. The central content area features a large heading 'Understand your dependencies' followed by a descriptive paragraph about dependency management. A search bar at the bottom left contains the placeholder 'Search for open source packages'. To the right, a sidebar lists various dependency statistics: npm packages (1.80M), Go modules (717k), Maven artifacts (427k), PyPI packages (325k), Cargo crates (72k), and NuGet packages (Coming soon). The footer features decorative network icons.

Open Source Insights is an experimental project by Google.

open/source/insights

About Documentation 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.

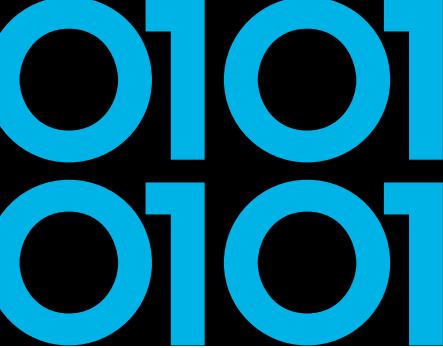
Search for open source packages

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



The screenshot shows a web browser window displaying the Deps.Dev interface for the npm package electron. The left panel lists package history entries:

Action	Version	Date
Version 17.0.0-alpha.4 added	17.0.0-alpha.4	November 25, 2021
Version 16.0.2 added	16.0.2	November 24, 2021
Default version changed from 16.0.1 to 16.0.2	16.0.2	November 24, 2021
Version 17.0.0-alpha.3 added	17.0.0-alpha.3	November 22, 2021
Version 16.0.1 added	16.0.1	November 18, 2021

Below the history is a note: "Package metadata as of December 1, 2021." The right panel displays the "OpenSSF scorecard" with the following results:

Check	Status
Branch-Protection	FAIL
CII-Best-Practices	FAIL
Code-Review	PASS
Fuzzing	FAIL
Packaging	UNKNOWN
Security-Policy	PASS
Signed-Releases	FAIL
Token-Permissions	PASS

Below the scorecard is a note: "Project metadata as of December 1, 2021."

Source Generators

0101
0101

A screenshot of a web browser window showing a blog post. The browser has a dark theme with a light-colored header bar. The address bar shows the URL <https://www.veracode.com/blog/secure-de...>. The page content is as follows:

Home | Blog | Secure Development

.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 dark blue header bar with the "Reproducible Builds" logo on the left. Below it is a light gray sidebar containing links: Home, Contribute, Documentation (which is bolded), Tools, Who is involved?, News, Events, and Talks. The main content area to the right has a large title "Definitions" and a sub-section "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.



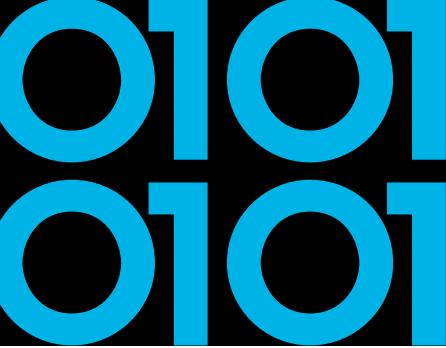
Reproducible/Deterministic Builds

- 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’

0101
0101

Reproducible Build .NET6

000001a0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000001a0:	0000 0000 0000 0000 0000 0000 0000 0000 0000
000001b0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000001b0:	0000 0000 0000 0000 ea03 0000 0000 0000 0000
000001c0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000001c0:	0000 0000 0000 0000 0000 0000 0000 0000 0000
000001d0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000001d0:	0000 0000 0000 0000 805a c352 00eb a154z.
000001e0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000001e0:	0000 0000 0000 0000 0000 0000 0000 0000 0000
000001f0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000001f0:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000200:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000200:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000210:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000210:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000220:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000220:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000230:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000230:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000240:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000240:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000250:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000250:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000260:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000260:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000270:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000270:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000280:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000280:	0000 0000 0000 0000 0000 0000 0000 0000 0000
00000290:	0000 0000 0000 0000 0000 0000 0000 0000 0000	00000290:	0000 0000 0000 0000 0000 0000 0000 0000 0000
000002a0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000002a0:	0000 0000 0000 0000 0000 0000 0000 0000 0000
000002b0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000002b0:	0000 0000 0000 0000 0000 0000 0000 0000 0000
000002c0:	0000 0000 0000 0000 0000 0000 0000 0000 0000	000002c0:	0000 0000 0000 0000 0000 0000 0000 0000 0000
000002d0:	0000 0000 0000 60e1 c552 6c5b 94d9	000002d0:	0000 0000 0000 b418 c252 0000 0000
000002e0:	2093 f53f c3d8 4298 8392 f53f dd07 20b5?	000002e0:	2c65 19e2 5817 f63f 2c65 19e2 5817 f63f	.e.X.?e.
000002f0:	8993 f53f 6d73 637a c292 f53f b81e 85eb	...?mscz...	000002f0:	bbb8 8d06 f016 f63f bbb8 8d06 f016 f63f?
00000300:	51b8 1d40 60e1 c552 0000 0000 a7e1 c552	Q. @`..R.	00000300:	0d00 0000 0000 0000 ef18 c252 0000 0000
00000310:	3485 ce6b ec92 f53f df37 bef6 cc92 f53f	4..k...?7.	00000310:	f018 c252 0000 0000 bbb8 8d06 f016 f63f	..R....
00000320:	6c43 c538 7f93 f53f 170e 8464 0193 f53f	1C.8...?...	00000320:	2c65 19e2 5817 f63f bbb8 8d06 f016 f63f	.e.X.?
00000330:	85eb 51b8 1e45 3040 a7e1 c552 0000 0000	..Q..E00...	00000330:	bbb8 8d06 f016 f63f 0900 0000 0000 0000?
00000340:	dde1 c552 89d2 dee0 0b93 f53f df4f 8d97	..R.....	00000340:	f018 c252 0000 0000 2c19 c252 0000 0000	..R....
00000350:	6e92 f53f dd07 20b5 8993 f53f c3d8 4298	n...?....	00000350:	bbb8 8d06 f016 f63f 2c65 19e2 5817 f63f?e.
00000360:	8392 f53f 5c8f c2f5 285c 1140 dde1 c552	...?\\...(V.	00000360:	bbb8 8d06 f016 f63f 2c65 19e2 5817 f63f?e.
00000370:	0000 0000 1fe2 c552 c3d8 4298 8392 f53f	...R..B.	00000370:	0800 0000 0000 0000 f118 c252 0000 0000
00000380:	c3d8 4298 8392 f53f c190 d5ad 9e93 f53f	..B...?...	00000380:	6819 c252 0000 0000 2c65 19e2 5817 f63f	h..R...e.
00000390:	c3d8 4298 8392 f53f 85eb 51b8 1e85 eb3f	..B...?..Q	00000390:	2c65 19e2 5817 f63f bbb8 8d06 f016 f63f	.e.X.?
000003a0:	1fe2 c552 0000 0000 55e2 c552 183e 22a6	..R...U..	000003a0:	bbb8 8d06 f016 f63f 0c00 0000 0000 0000?
000003b0:	4492 f53f 183e 22a6 4492 f53f 87a2 409f	D..?>"D..	000003b0:	f218 c252 0000 0000 a419 c252 0000 0000	R....
000003c0:	c893 f53f 183e 22a6 4492 f53f 14ae 47e1	...?>"D..	000003c0:	bbb8 8d06 f016 f63f bbb8 8d06 f016 f63f?
000003d0:	7a14 fe3f 55e2 c552 0000 0000 8ce2 c552	z..?U..R..	000003d0:	bbb8 8d06 f016 f63f bbb8 8d06 f016 f63f?



Reproducible/Deterministic Builds

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

Signing artifacts

0101
0101

A screenshot of a web browser displaying the Sigstore website at <https://www.sigstore.dev>. The page has a light beige background. At the top left is the Sigstore logo (a stylized 'f' icon) and the word 'sigstore'. To its right is a navigation bar with links: Overview (underlined), Community, How sigstore works, Trust and security, Blog, Docs (with a dropdown arrow), and a user icon. Below the navigation is a large, bold, dark gray headline: 'A new standard for signing, verifying and protecting software'. Underneath it is a smaller, dark gray subtext: 'Making sure your software's what it claims to be.' At the bottom, there's a section titled 'In collaboration with' featuring logos from various partners: ChainGuard, Cisco, Google, Hewlett Packard Enterprise, The Linux Foundation, Purdue University, Red Hat, and VMware.

Home · Sigstore

https://www.sigstore.dev

sigstore

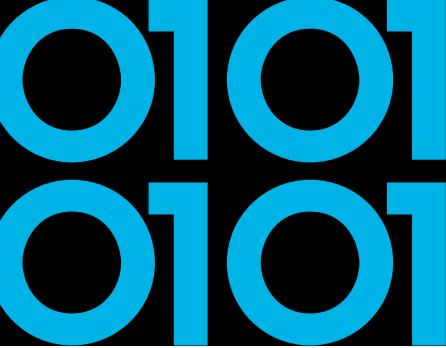
Overview Community How sigstore works Trust and security Blog Docs

A new standard for signing, verifying and protecting software

Making sure your software's what it claims to be.

In collaboration with

CHAIN GUARD CISCO Google Hewlett Packard Enterprise THE LINUX FOUNDATION PURDUE UNIVERSITY Red Hat 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

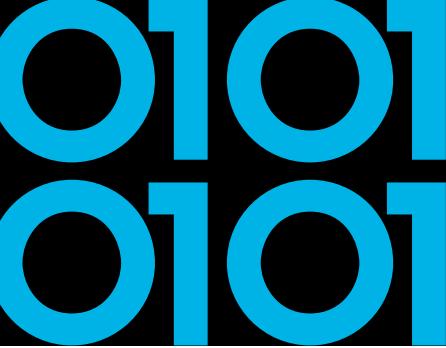
This means that open source software uploaded for distribution has a stricter, more standardized way of checking who's been involved, that it hasn't been tampered with. There's no risk of key compromise, so third parties can't hijack a release and slip in something malicious.

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"] --> SignAndPublish["SIGN AND PUBLISH ARTIFACTS"]; Developers --> PublishCertificates["PUBLISH SIGNING CERTIFICATES"]; Developers --> MonitorLogs["MONITOR LOGS"]; SignAndPublish --> Fulcio["FULCIO CERTIFICATE AUTHORITY"]; PublishCertificates --> SignatureLog["SIGNATURE TRANSPARENCY LOG"]; MonitorLogs --> 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 icons representing "SIGN AND PUBLISH ARTIFACTS", "PUBLISH SIGNING CERTIFICATES", and "MONITOR LOGS". Dotted arrows point from each of these icons down to three corresponding boxes below: "FULCIO CERTIFICATE AUTHORITY", "SIGNATURE TRANSPARENCY LOG", and "KEY TRANSPARENCY LOG". Finally, dotted arrows point from each of these three boxes down to a large orange box at the bottom labeled "TRUST ROOT".

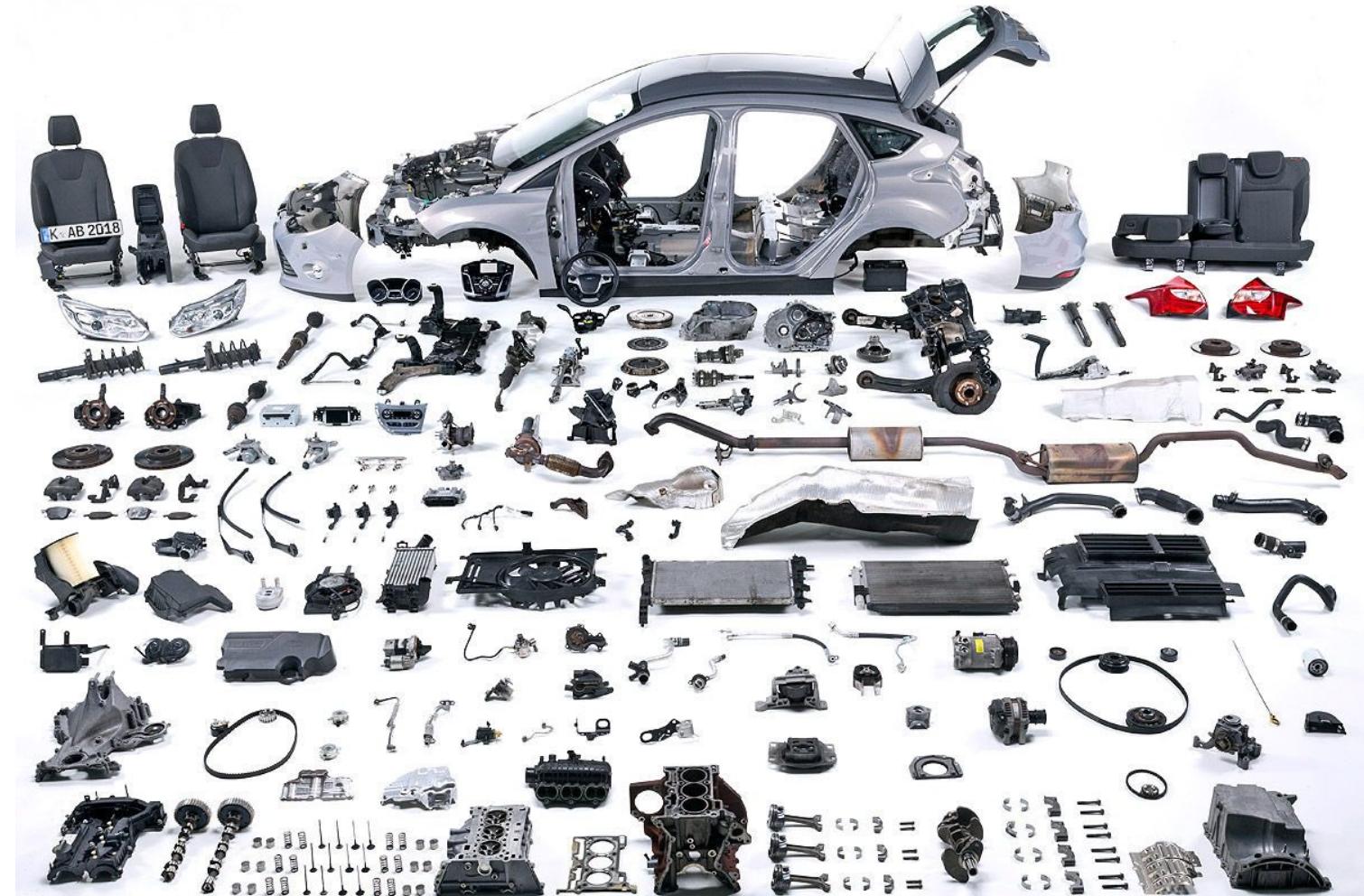


Signing artifacts

- Cosign can do keyless signing based on OpenID Connect
- GitHub Actions have released OpenID Connect support
- Demo time!

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

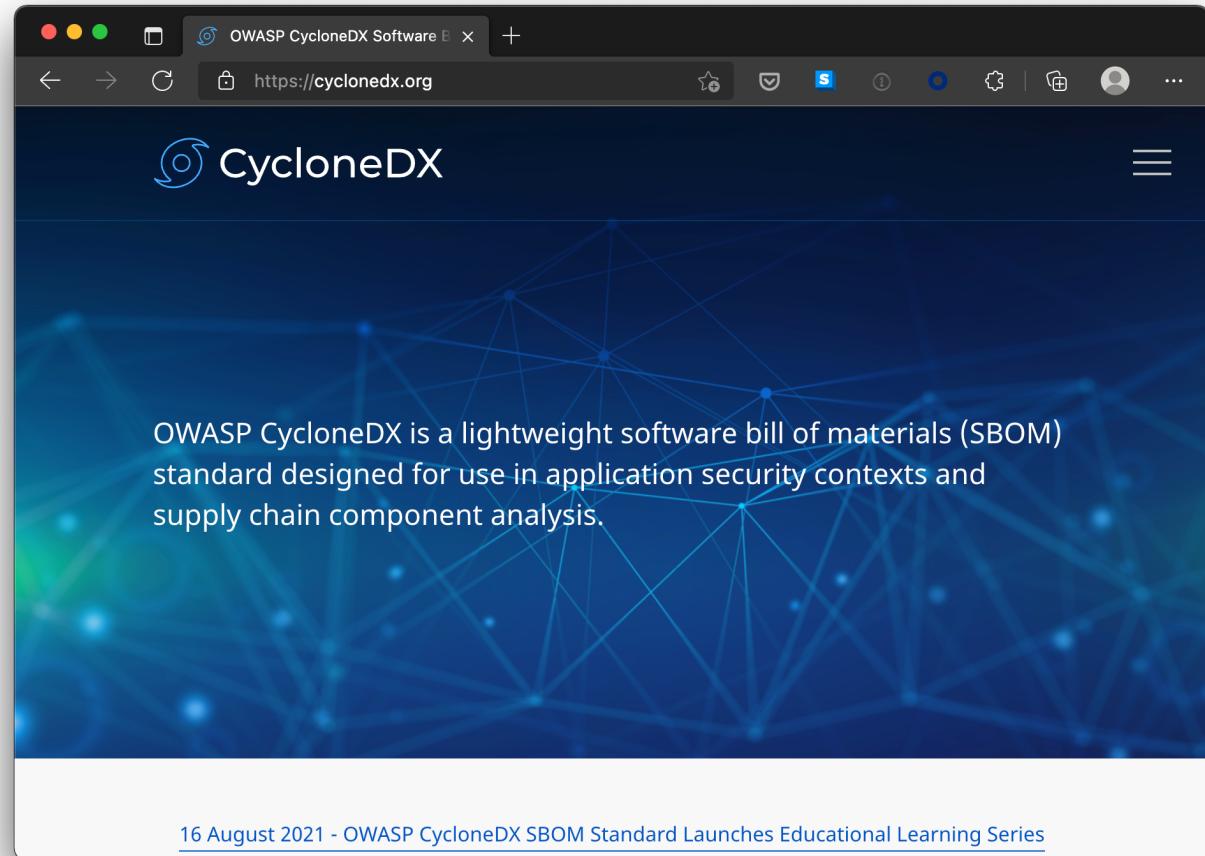


Software Bill of Materials (SBOM)

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

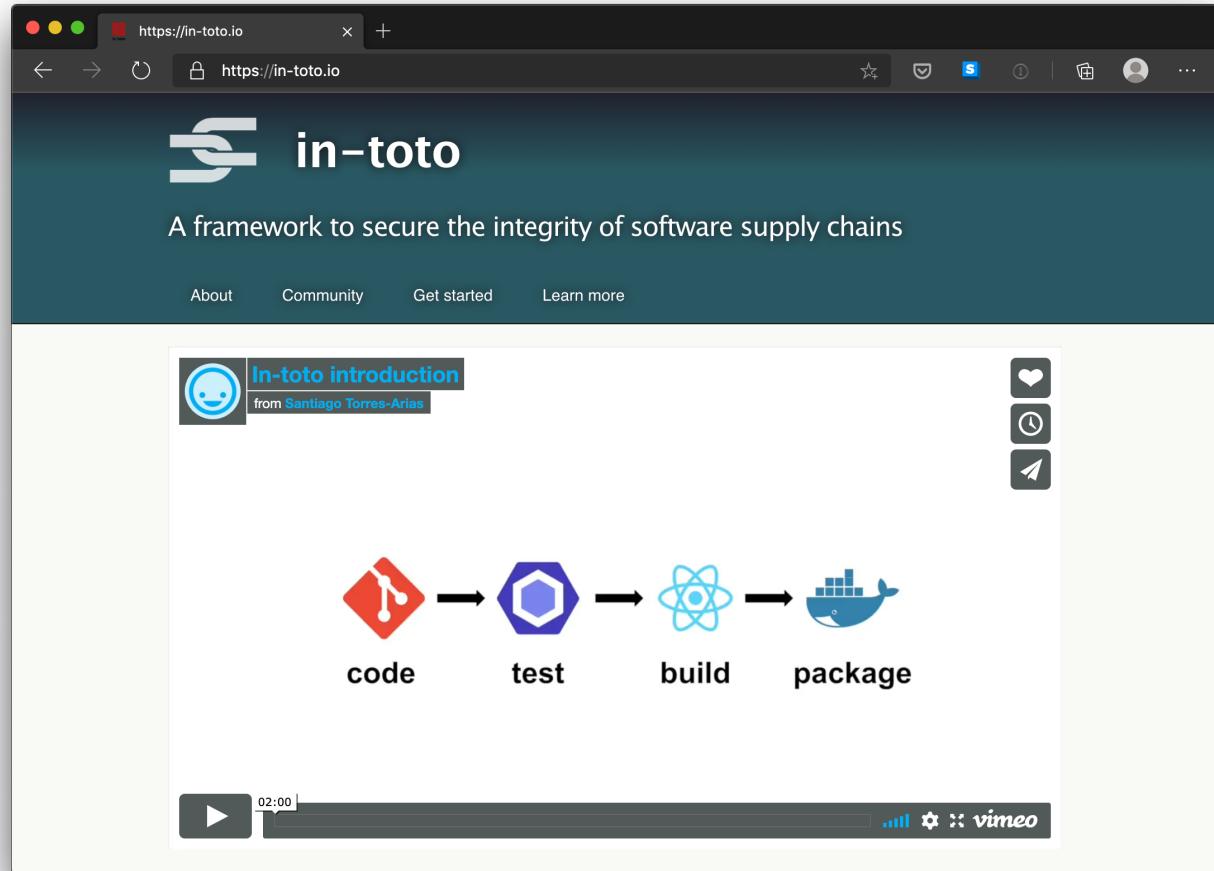
0101
0101

Software Bill of Materials (SBOM)



0101
0101

In-toto





In-Toto - Demo - 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

0101
0101

In-Toto - Demo

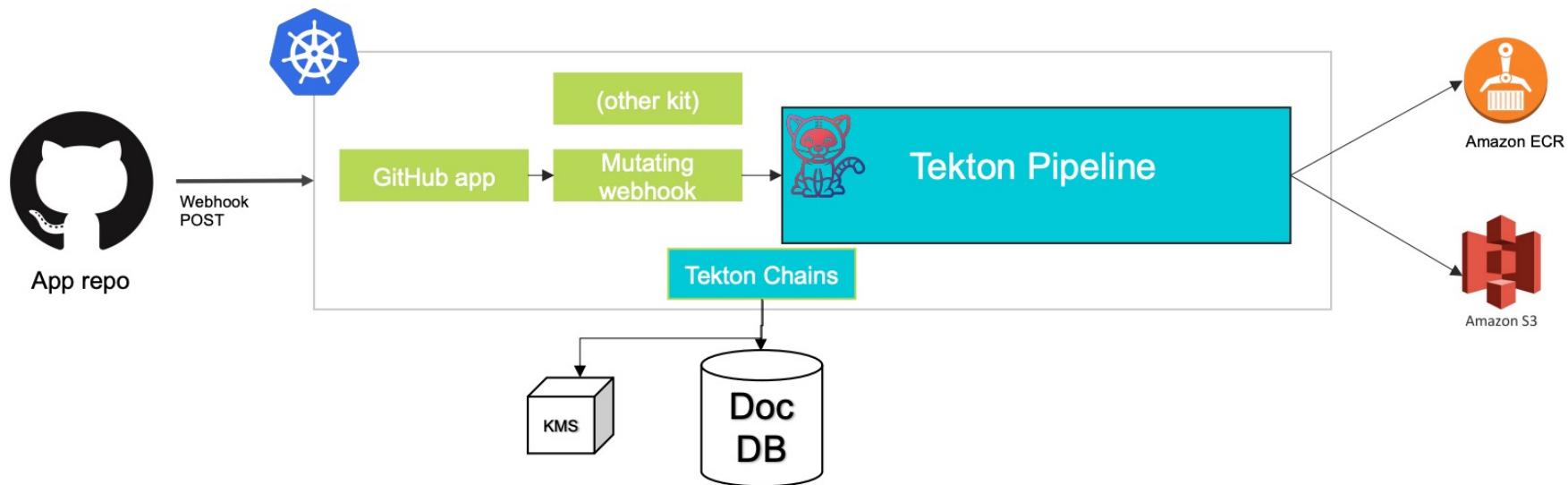




SolarWinds Project Trebuchet



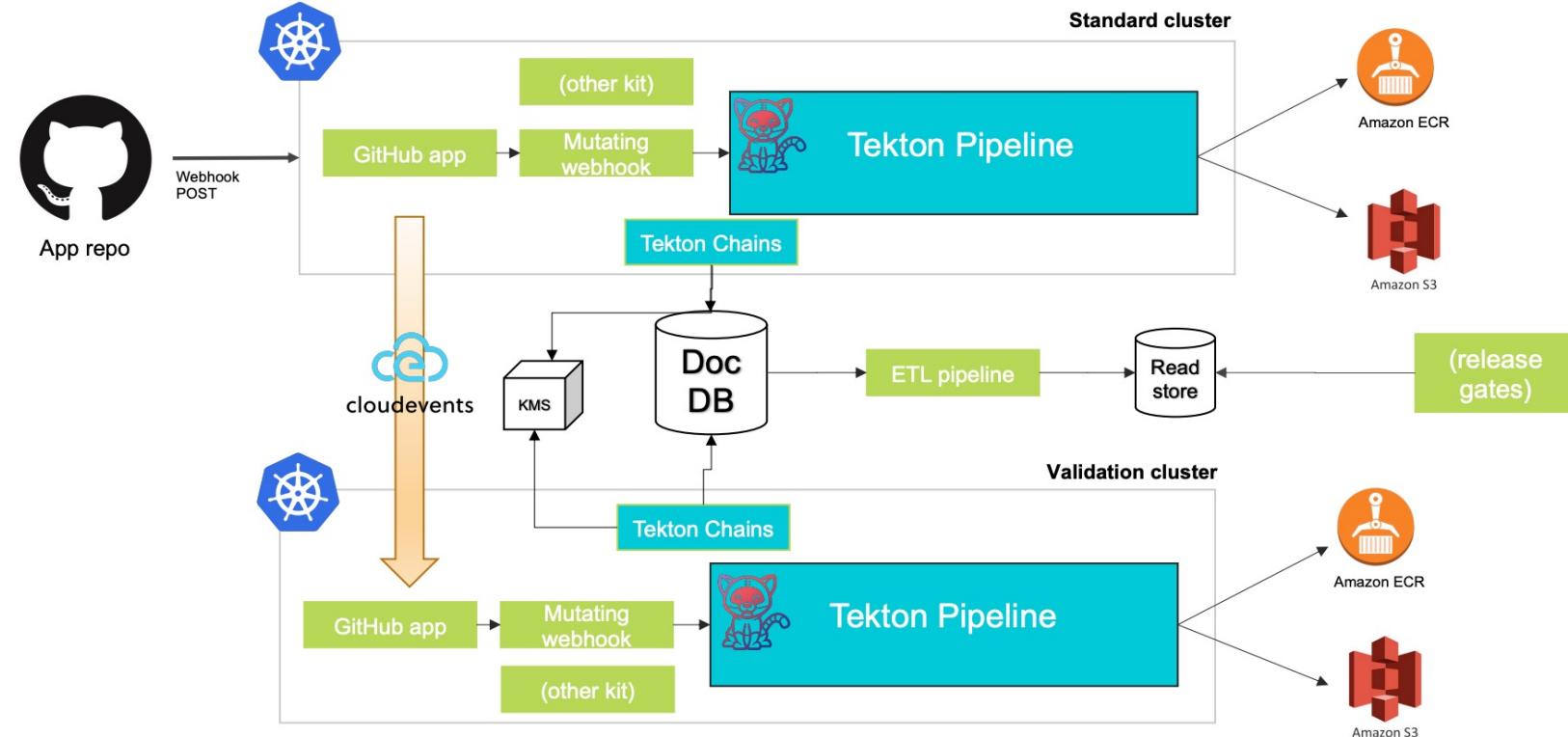
Pipeline With Attestations



SolarWinds Project Trebuchet



Reading Results



IBM OpenShift

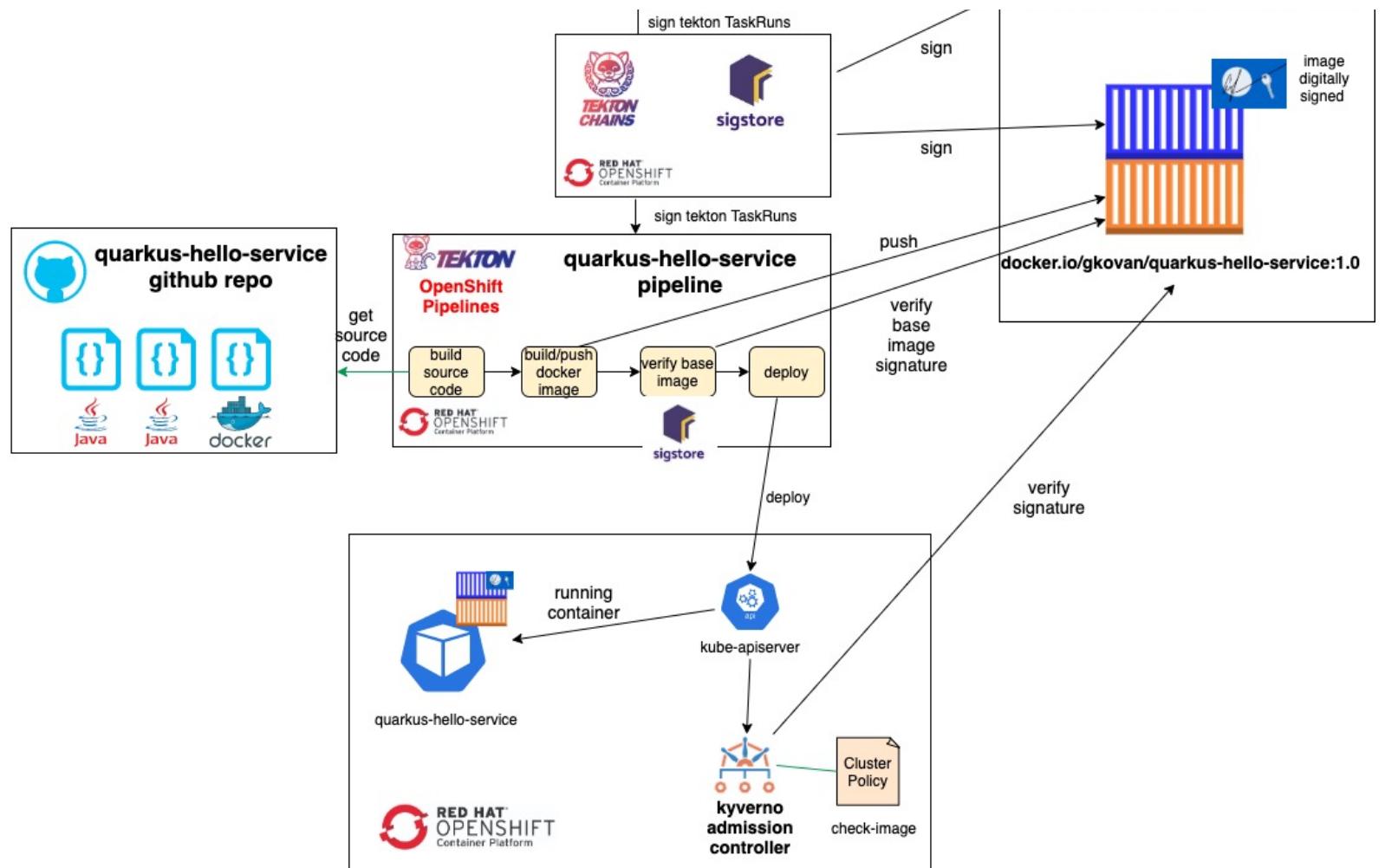
0101
0101

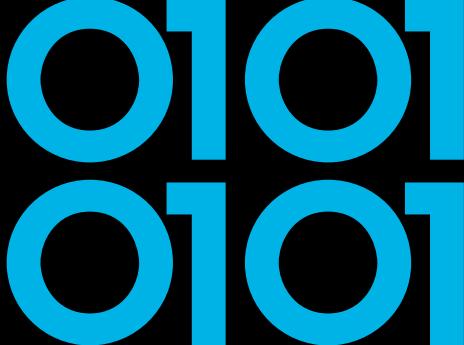
A screenshot of a web browser displaying a Medium article. The article title is "A Zero Trust Approach for Securing the Supply Chain of Microservices Packaged as Container Images" by Gerry Kovan. The browser interface shows standard navigation controls, a URL bar with the address <https://gkovanc.medium.com/a-zero-trust-approach-for-securing-the-sup...>, and various browser extension icons.

The article author is Gerry Kovan, with 28 Followers. There are buttons for "Follow" and "Get started". The main content area features the article title in large, bold, dark font. Below the title, it says "Sep 6 · 6 min read" and includes social sharing icons for Twitter, Facebook, LinkedIn, and others.

The article summary begins with: "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

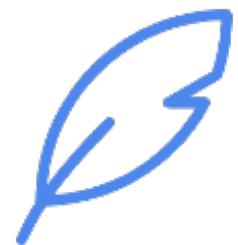
IBM OpenShift





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 Microsoft Docs page titled "Artifact policy checks" for Azure Pipelines. The page is part of the "Documentation" section under "Azure DevOps". The URL is <https://docs.microsoft.com/en-us/azure/devops/pipelines/process/artifact-pol...>. The page content includes a sidebar for "Version" (set to "Azure DevOps Services") and "Filter by title". The main article discusses "Artifact policy checks" and their purpose in enforcing policies before deployment to critical environments like production. It also mentions "Creating custom policies". The page footer includes standard Microsoft documentation links.

Artifact policy checks

11/05/2019 • 2 minutes to read • 11 contributors +1

In this article

Prerequisites

Creating custom policies

Artifact policies are enforced before deploying to critical environments such as production. These policies are evaluated against all the deployable artifacts in the given pipeline run and block the deployment if the artifacts don't comply. Adding a check to evaluate Artifact requires

Google SLSA

0101
0101

The screenshot shows a web browser window with the title bar "Overview | SLSA". The address bar displays the URL "https://slsa.dev". The page content is as follows:

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



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.
- Integrate security into your software lifecycle.
- Start working on creating SBOM's and see how SLSA can fit into your process.

VERACODE

Thanks! Questions?

<https://github.com/nielstanis/ndcoslo2021>

<https://github.com/nielstanis/ndcoslo2021-webapp>

ntanis at veracode.com

@nielstanis on Twitter

