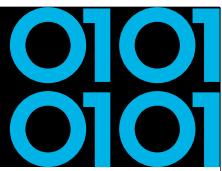


NDC { Sydney }

Securing your .NET application
software supply-chain

Niels Tanis





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



NDC { Sydney }

@nielstanis

Securing your .NET application software supply-chain

0101
0101

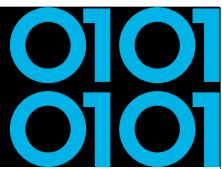


NDC { Sydney }

@nielstanis

Picture is from Veracode report/site:

<https://www.veracode.com/sites/default/files/pdf/resources/papers/everything-you-need-to-know-open-source-risk/index.html>

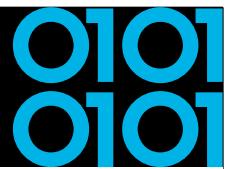


Agenda

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

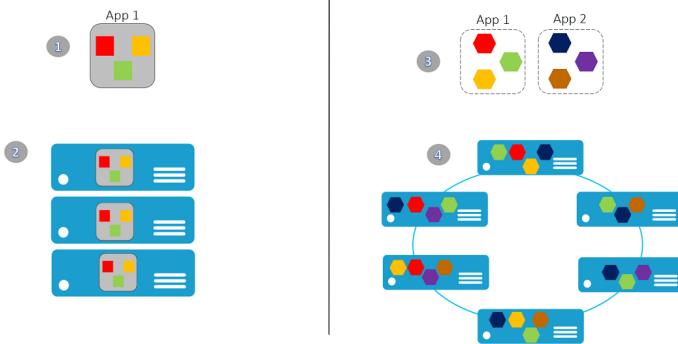
NDC { Sydney }

@nielstanis



Evolution in Software Architecture

- Monolith
- Microservices
- Serverless
- Cloud-Native



NDC { Sydney }

@nielstanis

0101
0101

What is a Supply Chain?



NDC { Sydney }

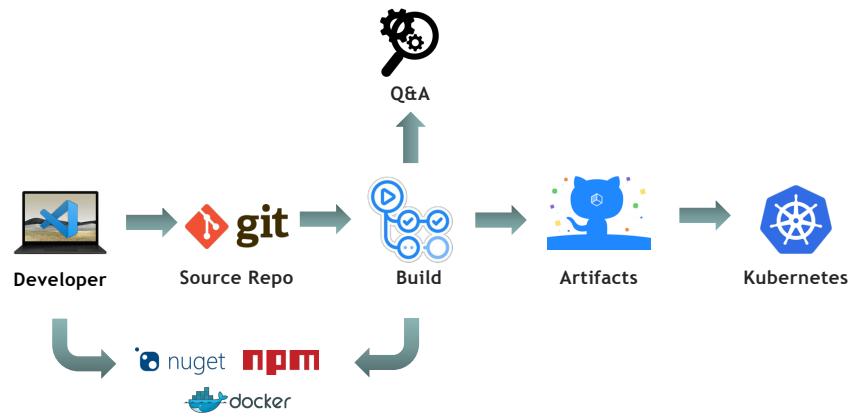
@nielstanis

Image source:

https://www.wardsauto.com/sites/wardsauto.com/files/styles/article_featured_retina/public/Renault%20Kadjar%20assembly%20line%20-%20Palencia%20Spain-5_8.jpg?

0101
0101

Software Supply Chain

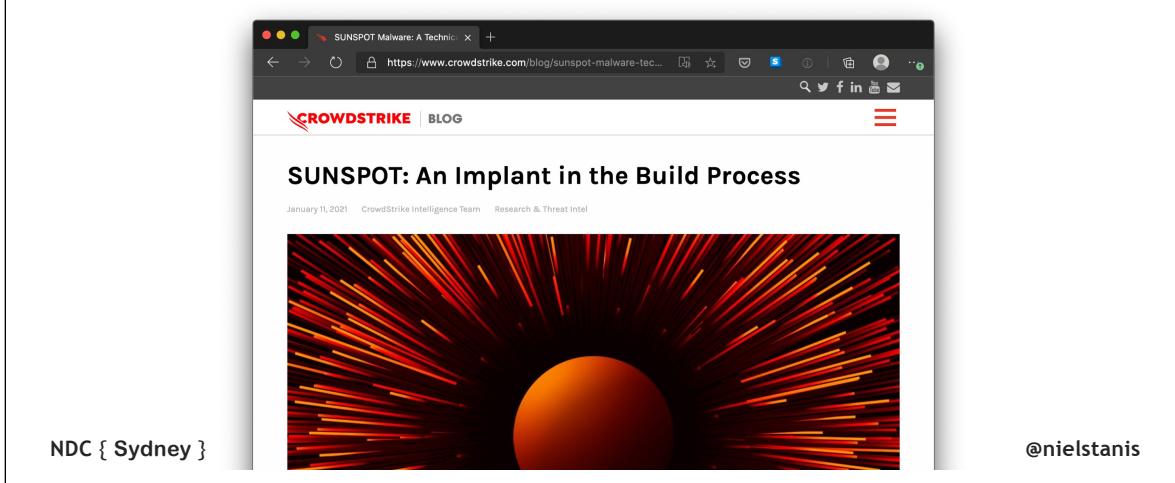


NDC { Sydney }

@nielstanis

SolarWinds SunSpot

0101
0101

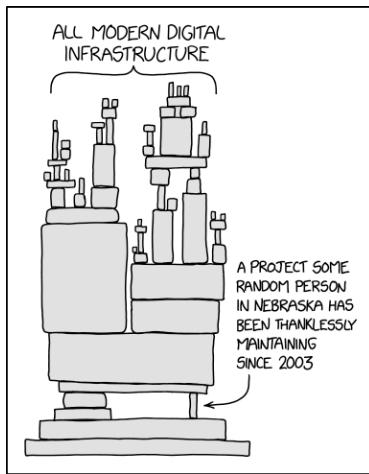


<https://www.crowdstrike.com/blog/sunspot-malware-technical-analysis/>

<https://www.microsoft.com/security/blog/2020/12/18/analyzing-solorigate-the-compromised-dll-file-that-started-a-sophisticated-cyberattack-and-how-microsoft-defender-helps-protect/>

0101
0101

XKDC - Dependency



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

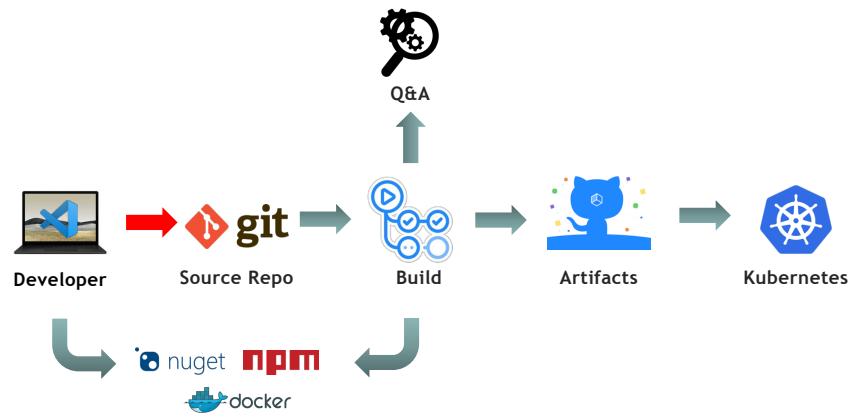
NDC { Sydney }

@nielstanis

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

0101
0101

Software Supply Chain



NDC { Sydney }

@nielstanis

0101
0101

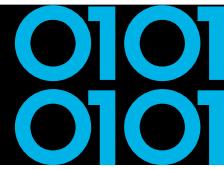
GitHub account

The screenshot shows a news article from ZDNet. The title is "Canonical GitHub account hacked, Ubuntu source code safe". The subtitle reads, "Ubuntu source code appears to be safe; however Canonical is investigating." Below the article is a screenshot of the Canonical GitHub profile page, which shows a message from a user named "CAN_ISOT_MAKED_50" stating, "I'm sorry, but I'm not interested in your request." At the bottom right of the screenshot, there is a link: "Why everyone should have this cheap security tool!"

NDC { Sydney }

@nielstanis

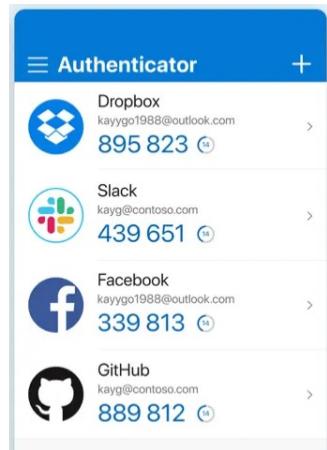
<https://www.zdnet.com/article/canonical-github-account-hacked-ubuntu-source-code-safe/>



Slide with larger header
when there is not a
lot of text heavy

The content text slide
bullets as preview, h
you do not need to
bullets. Text is prese
Trebuchet size 14
bullets may be rem
not needed.

Use MFA on source-repository



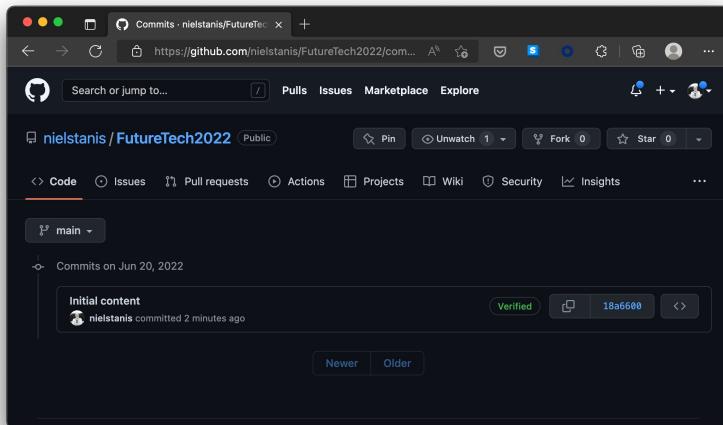
NDC { Sydney }

@nielstanis

<https://help.github.com/en/github/authenticating-to-github/configuring-two-factor-authentication>

GIT Commit Signing

0101
0101



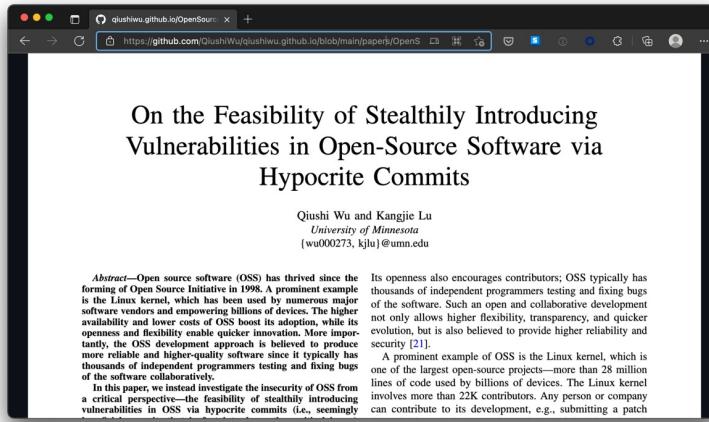
NDC { Sydney }

@nielstanis

<https://www.hanselman.com/blog/HowToSetupSignedGitCommitsWithAYubiKeyNEOAndGPGAndKeybaseOnWindows.aspx>

Hypocrite Commits

0101
0101

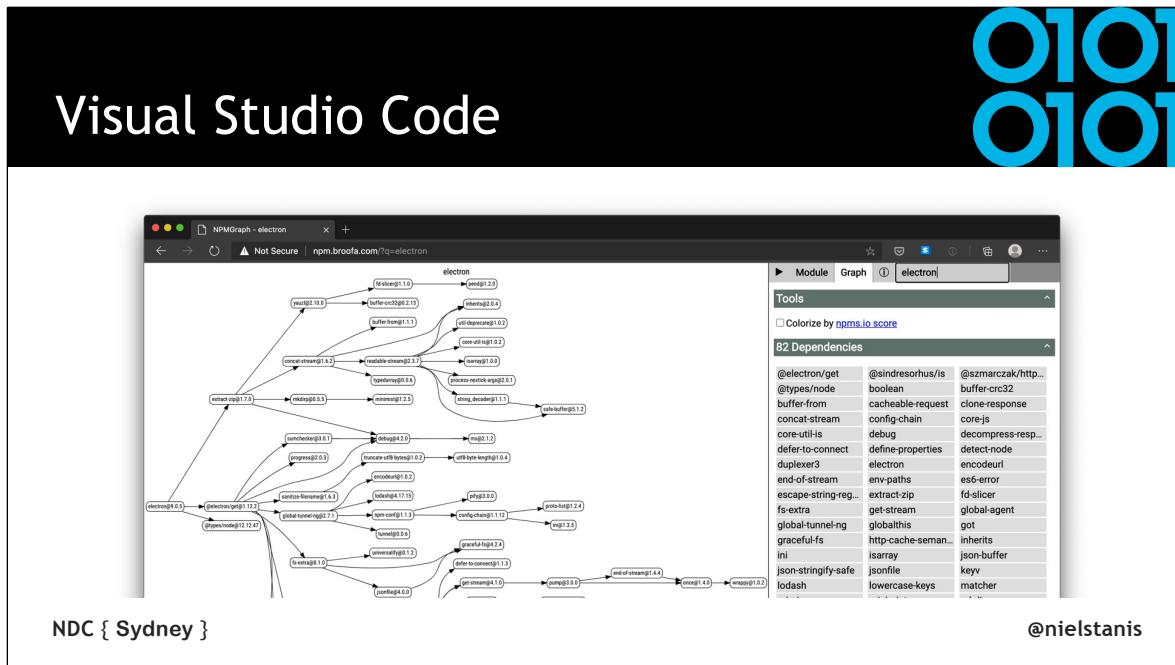


NDC { Sydney }

@nielstanis

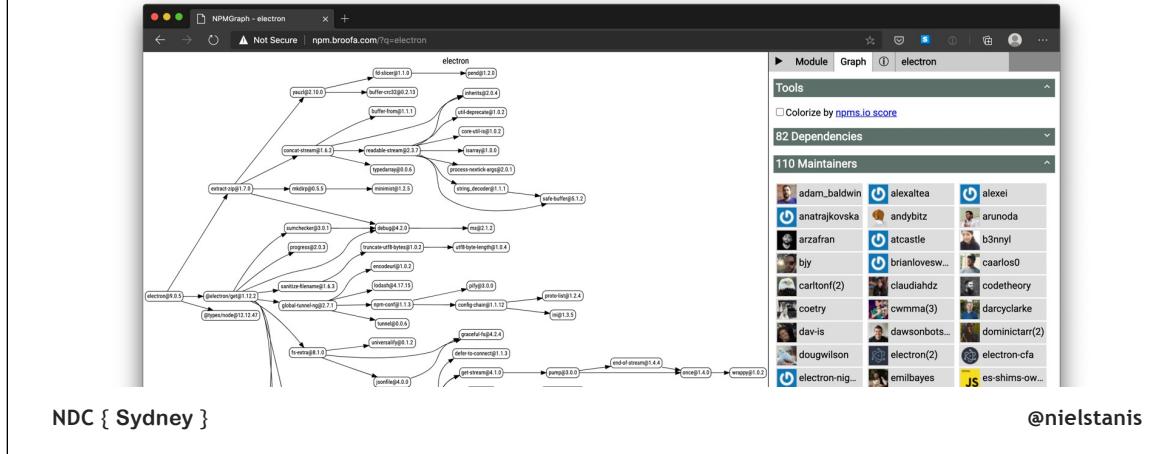
<https://github.com/QiushiWu/qiushiwu.github.io/blob/main/papers/OpenSourceInsecurity.pdf>

Visual Studio Code



0101
0101

Visual Studio Code



NDC { Sydney }

@nielstanis

Visual Studio Code

0101
0101

The screenshot shows a Microsoft Edge browser window with the following details:

- Page Title: CVE-2022-30129 - Security Update Guide
- URL: https://msrc.microsoft.com/update-guide/en-US/vulnerabilities/CVE-2022-30129
- Section: Visual Studio Code Remote Code Execution Vulnerability
- Identifier: CVE-2022-30129
- Release Date: May 10, 2022
- Assigning CNA: Microsoft
- CVSS3.1 Score: 8.8 / 7.7

NDC { Sydney }

@nielstanis

<https://msrc.microsoft.com/update-guide/vulnerability/CVE-2022-30129>

Visual Studio Code

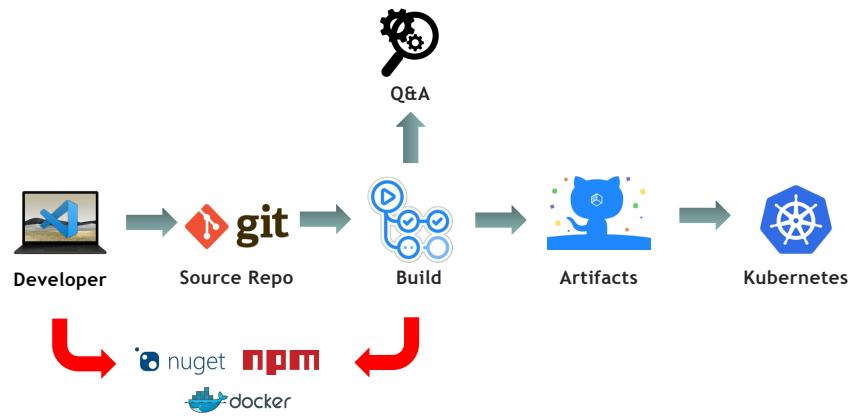
0101
0101



<https://www.bleepingcomputer.com/news/security/heres-how-a-researcher-broke-into-microsoft-vs-codes-github/>

0101
0101

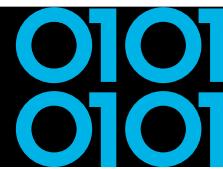
3rd Party Libraries



NDC { Sydney }

@nielstanis

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."*



NDC { Sydney }

@nielstanis

<https://info.veracode.com/fy22-state-of-software-security-v11-open-source-edition.html>

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-2022-24512 | .NET Remote Code Execution Vulnerability #213". The issue was opened by `dchwittaker` on March 8th, 2022, and has 0 comments. The issue details section contains the following text:

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

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](#)

The right sidebar shows the following details:

- 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

NDC { Sydney }

@nielstanis

<https://github.com/dotnet/announcements/issues/213>

0101
0101

Vulnerabilities in libraries



Alerts and Tips Resources Industrial Control Systems

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

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

Original release date: October 22, 2021

[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 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](#).

NDC { Sydney }

@nielstanis

<https://us-cert.cisa.gov/ncas/current-activity/2021/10/22/malware-discovered-popular-npm-package-ua-parser-js>

<https://portswigger.net/daily-swig/popular-npm-package-ua-parser-js-poisoned-with-cryptomining-password-stealing-malware>

Vulnerabilities in libraries

0101
0101

The image shows two side-by-side screenshots of GitHub blog posts. The left screenshot is titled "GitHub's commitment to npm ecosystem security" by Mike Hanley on November 15, 2021. It features a cartoon illustration of a character working on a large puzzle piece. The right screenshot is titled "Enrolling all npm publishers in enhanced login verification and next steps for two-factor authentication enforcement" by the NPM team on December 7, 2021. It features the NPM logo. At the bottom left is the text "NDC { Sydney }" and at the bottom right is the handle "@nielstanis".

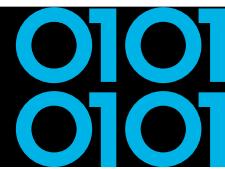
<https://github.blog/2021-11-15-githubs-commitment-to-npm-ecosystem-security/>
<https://github.blog/2021-12-07-enrolling-npm-publishers-enhanced-login-verification-two-factor-authentication-enforcement/>

0101
0101

Dependency Confusion

The screenshot shows a web browser window displaying 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. Below the title, it says "The Story of a Novel Supply Chain Attack". The author's profile picture and name are visible, along with sharing icons for Twitter, Facebook, LinkedIn, and others. A large image of colorful shipping containers is shown below the title. At the bottom left of the screenshot, there is a watermark that reads "NDC { Sydney }". At the bottom right, there is a handle "@nielstanis".

<https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610>



Microsoft Whitepaper

3 ways to mitigate risk when using private package feeds

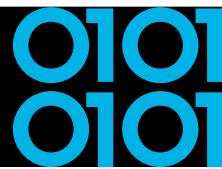
Secure Your Hybrid Software Supply Chain

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

NDC { Sydney }

@nielstanis

<https://azure.microsoft.com/nl-nl/resources/3-ways-to-mitigate-risk-using-private-package-feeds/>
<https://azure.microsoft.com/mediahandler/files/resourcefiles/3-ways-to-mitigate-risk-using-private-package-feeds/3%20Ways%20to%20Mitigate%20Risk%20When%20Using%20Private%20Package%20Feeds%20-%20v1.0.pdf>



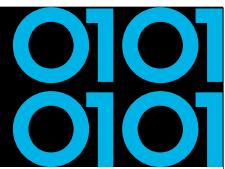
Dependency Confusion - NuGet

- 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

NDC { Sydney }

@nielstanis

<https://azure.microsoft.com/nl-nl/resources/3-ways-to-mitigate-risk-using-private-package-feeds/>
<https://azure.microsoft.com/mediahandler/files/resourcefiles/3-ways-to-mitigate-risk-using-private-package-feeds/3%20Ways%20to%20Mitigate%20Risk%20When%20Using%20Private%20Package%20Feeds%20-%20v1.0.pdf>



3rd Party Libraries

- Intent of library, know what's inside!
- Keep in mind that's a transitive list of dependencies
- My talk 'Sandboxing .NET Assemblies' Tomorrow @ 3PM in Room 4
- Open Source Security Foundation - OpenSSF
- Security Scorecards - Security health metrics for Open Source

NDC { Sydney }

@nielstanis

[Sandboxing .NET assemblies for fun, profit and of course security! - Niels Tanis - NDC Porto 2022 - YouTube](#)

Security Scorecards - OpenSSF



The screenshot shows the GitHub repository for ossf/scorecard: Security Score. The README.md page is displayed. It features a large cartoon character of a blue bird-like creature holding a yellow shield with a '10' on it. The page has two main sections: 'Overview' and 'Using Scorecards'. The 'Overview' section includes links to 'What Is Scorecards?', 'Prominent Scorecards Users', and 'Scorecards Public Data'. The 'Using Scorecards' section lists 'Scorecards GitHub Action', 'Scorecards REST API', 'Scorecards Badges', 'Scorecards Command Line Interface' (with sub-links for Prerequisites, Installation, Authentication, and Basic Usage), and 'Scorecards Reference'.

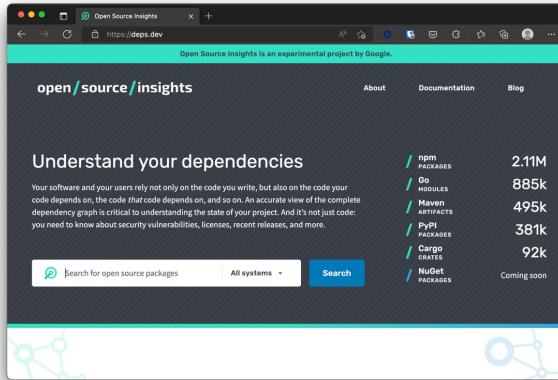
NDC { Sydney }

@nielstanis

<https://github.com/ossf/scorecard>

Deps.Dev by Google

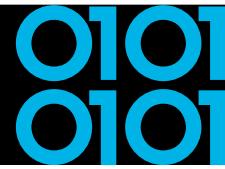
0101
0101



NDC { Sydney }

@nielstanis

<https://deps.dev/>



Deps.Dev by Google

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

6.6/10

Scorecard as of September 12, 2022.

➤ Maintained	10/10
➤ Code-Review	8/10
➤ CI-Best-Practices	0/10
➤ Vulnerabilities	10/10
➤ Security-Policy	10/10
➤ Dangerous-Workflow	10/10

➤ Token-Permissions	0/10
➤ License	10/10
➤ Pinned-Dependencies	7/10
➤ Binary-Artifacts	10/10
➤ Fuzzing	0/10
➤ Dependency-Update-Tool	10/10
➤ Signed-Releases	0/10
➤ SAST	0/10
➤ Branch-Protection	8/10

Project metadata as of September 18, 2022.

NDC { Sydney }

@nielstanis

<https://deps.dev/npm/electron>

Source Generators

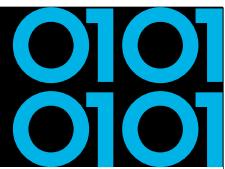
0101
0101

The screenshot shows a blog post on a dark-themed website. The title is ".NET 5, Source Generators, and Supply Chain Attacks". It's written by Mateusz Kreszowicz on September 30, 2021. The post discusses IDEs and build infrastructure as targets for threat actors, mentioning XcodeGhost and its malware-ridden Apple Xcode IDE. Below the article, there's a "Share this article" section with links to Twitter, Facebook, and LinkedIn. The URL of the post is https://www.veracode.com/blog/secure-development/net-5-source-generators-and-supply-chain-attacks.

NDC { Sydney }

@nielstanis

<https://www.veracode.com/blog/secure-development/net-5-source-generators-and-supply-chain-attacks>



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>
```

NDC { Sydney }

@nielstanis

Reproducible/Deterministic Builds

0101
0101



Home

Contribute

Documentation

Tools

Who is involved?

News

Events

Talks

Definitions

When is a build reproducible?

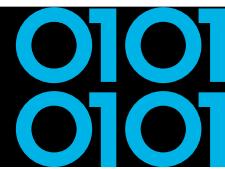
A build is **reproducible** if given the same source code, build environment and build instructions, any party can recreate bit-by-bit identical copies of all specified artifacts.

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.

NDC { Sydney }

@nielstanis

<https://reproducible-builds.org/docs/definition/>



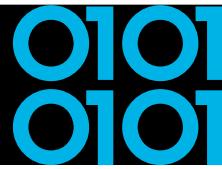
Reproducible/Deterministic Builds

- 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’

NDC { Sydney }

@nielstanis

<https://blog.paranoidcoding.com/2016/04/05/deterministic-builds-in-roslyn.html>
<https://github.com/dotnet/roslyn/blob/master/docs/compilers/Deterministic%20Inputs.md>
<https://github.com/clairernovotny/DeterministicBuilds>

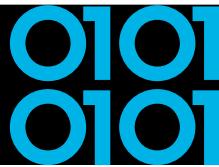


Reproducible Build .NET6

NDC { Sydney }

@nielstanis

<https://www.tabsoverspaces.com/233662-changing-paths-in-pdb-files-for-source-files-and-pdb-file-path-in-dll-as-well>
<DebugOutput> in CSPROJ demo



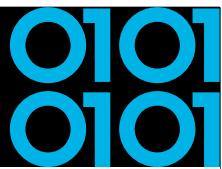
Reproducible/Deterministic Builds

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

NDC { Sydney }

@nielstanis

<https://blog.paranoidcoding.com/2016/04/05/deterministic-builds-in-roslyn.html>
<https://github.com/dotnet/roslyn/blob/master/docs/compilers/Deterministic%20Inputs.md>
<https://devblogs.microsoft.com/dotnet/producing-packages-with-source-link/>
<https://github.com/dotnet/reproducible-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`

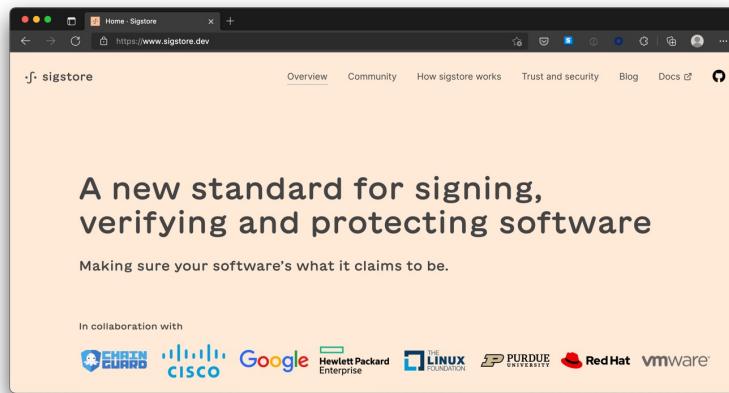
NDC { Sydney }

@nielstanis

<https://github.com/dotnet/designs/blob/main/accepted/2020/reproducible-builds.md>

Signing artifacts

0101
0101



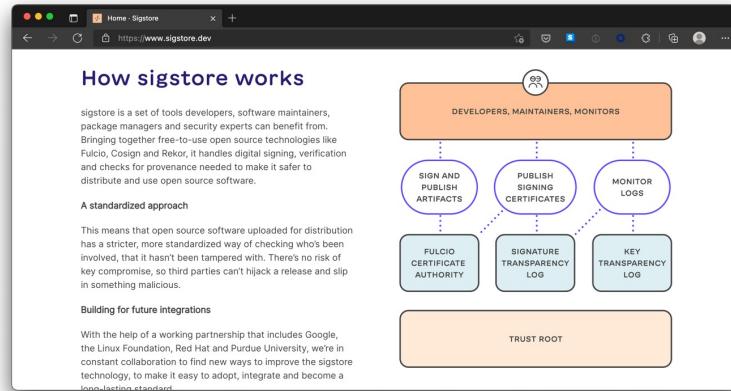
NDC { Sydney }

@nielstanis

<https://sigstore.dev>

0101
0101

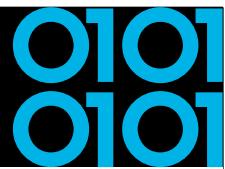
Signing artifacts



NDC { Sydney }

@nielstanis

<https://sigstore.dev>



Signing artifacts

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

NDC { Sydney }

@nielstanis

<https://sigstore.dev>

0101
0101

Cosign Keyless Signing



NDC { Sydney }

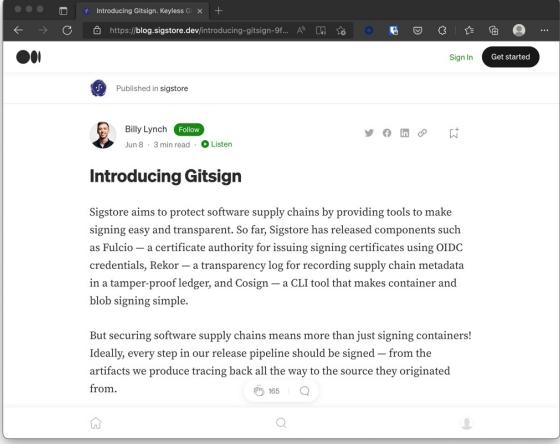
@nielstanis

<https://sigstore.dev>

<https://blog.sigstore.dev/introducing-gitsign-9fd3f1b682aa>

Git Commit Signing Sigstore GitSign

0101
0101



NDC { Sydney }

@nielstanis

<https://sigstore.dev>

<https://blog.sigstore.dev/introducing-gitsign-9fd3f1b682aa>

0101
0101

Automotive Industry



NDC { Sydney }

@nielstanis

0101
0101

Car Supply Chain



Tata Steel Factory

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

Bosch Factory

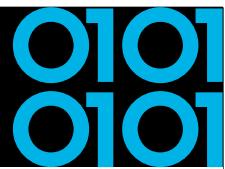
- Steel Batch #1234 Tata
- ECE-R90 Tested/Certified
 - Serie #45678
- Used by Ford, Volkswagen and Renault

Renault Manufacturing

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

NDC { Sydney }

@nielstanis



Software Bill of Materials (SBOM)

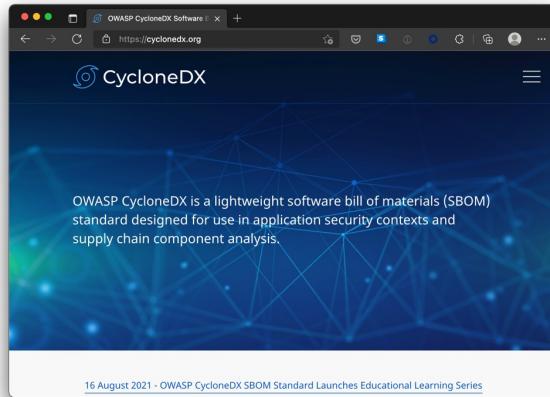
- Industry standard of describing the software
 - Producer Identity - Who Created it?
 - Product Identity - What's the product?
 - Integrity - Is the project unaltered?
 - Licensing - How can the project be used?
 - Creation - How was the product created? Process meets requirements?
 - Materials - How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

NDC { Sydney }

@nielstanis

Software Bill of Materials (SBOM)

0101
0101



NDC { Sydney }

@nielstanis

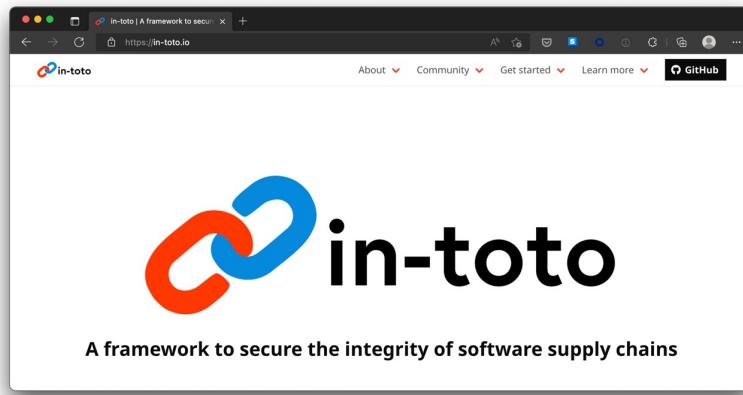
<https://cyclonedx.org>

The screenshot shows a blog post on the Docker website. The header features a large blue 'Docker SBOM' title and a blue '0101' graphic. The main content is a blog post titled 'Announcing Docker SBOM: A step towards more visibility into Docker images' by Justin Cormack, published on April 7, 2022. The post discusses the introduction of the 'docker sbom' command in Docker Desktop 4.7.0. The sidebar includes links for 'Products', 'Developers', 'Pricing', 'Blog', 'About Us', 'Partners', a search bar, and a 'Get Started' button. It also lists 'Post Tags' like 'docker', 'Docker images', 'sbom', and 'Secure Software Supply Chain', and 'Categories' such as 'Community', 'Company', 'Engineering', 'Newsletters', and 'Products'. The URL in the browser is <https://www.docker.com/blog/announcing-docker-sbom-a-step-towards-more-visibility-into-docker-images/>. A watermark 'NDC { Sydney }' is visible on the left, and a handle '@nielstanis' is on the right.

<https://www.docker.com/blog/announcing-docker-sbom-a-step-towards-more-visibility-into-docker-images/>

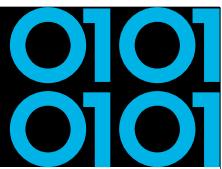
In-toto

0101
0101



NDC { Sydney }

@nielstanis



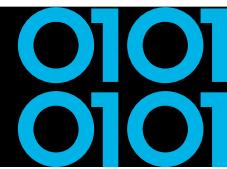
In-Toto - Terminology

- **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

NDC { Sydney }

@nielstanis

<https://in-toto.io/>



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

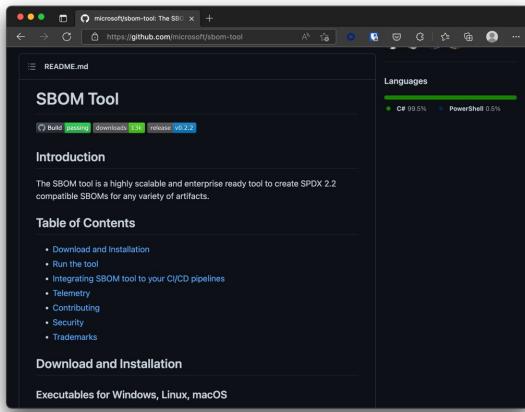
NDC { Sydney }

@nielstanis

<https://youtu.be/fYCfB7MZPh4?t=2777>

Microsoft SBOM Tool

0101
0101



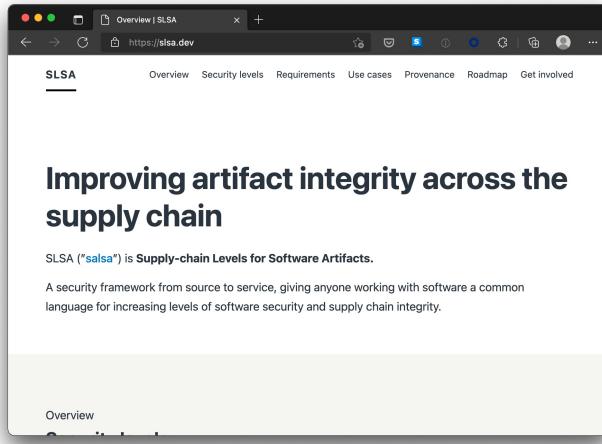
NDC { Sydney }

@nielstanis

<https://github.com/microsoft/sbom-tool>

Google SLSA

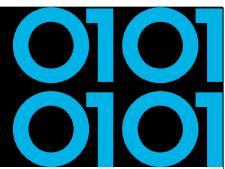
0101
0101



NDC { Sydney }

@nielstanis

<https://slsa.dev>



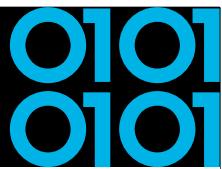
Google SLSA Levels

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

NDC { Sydney }

@nielstanis

<https://slsa.dev>



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

NDC { Sydney }

@nielstanis

<https://security.googleblog.com/2022/04/improving-software-supply-chain.html>

SLSA3 Generator GitHub Actions

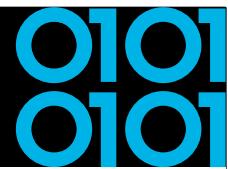
0101
0101

The screenshot shows a web browser displaying a blog post from the SLSA website. The title of the post is "General availability of SLSA3 Generic Generator for GitHub Actions". The post is authored by Ian Lewis, Laurent Simon, Asra Ali, and was published on 29 Aug 2022. The content of the post discusses the release of a Go builder and the addition of a new tool for generating provenance documents for projects developed in any programming language. The URL of the post is <https://slsa.dev/blog/2022/08/slsa-github-workflows-generic-ga>.

NDC { Sydney }

@nielstanis

<https://slsa.dev/blog/2022/08/slsa-github-workflows-generic-ga>



MyAwesomePDFComponent Demo

- A NuGet Package build in GitHub Actions
- CycloneDX SBOM
- Sigstore Keyless Signing
- SLSA Level 3 Provenance
- SBOM data, know what?

NDC { Sydney }

@nielstanis

SUSE SLSA Level 4

0101
0101

The screenshot shows a web browser displaying a document titled "SLSA: Securing the Software Supply Chain". The page has a dark header with the SUSE logo and navigation links. The main content area features a green header bar with the title "SLSA: Securing the Software Supply Chain". Below this, there's an "Abstract" section and a "Disclaimer" section. The "Abstract" section contains a brief summary of the document's purpose. The "Disclaimer" section provides legal disclaimers about the document's origin and accuracy. At the bottom right of the page, there are social media sharing icons.

NDC { Sydney }

@nielstanis

<https://documentation.suse.com/sbp/server-linux/html/SBP-SLSA4/index.html>

0101
0101

Witness & GitLab Attestator

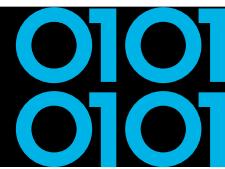
The screenshot shows a GitLab repository named 'Witness Demo'. The repository has 44 commits, 5 branches, and 0 tags. The main branch contains a file named 'witness-runner-working' which was authored by Nicholas Kennedy 1 month ago. The repository also includes a 'README' file and a 'CICD configuration' file. A note indicates 'No license. All rights reserved.' Below the repository details, there is a table showing the last commit for each file:

Name	Last commit	Last update
policy	Update policy/gcp.rego	6 months ago
.dockergignore	Initial demo code	10 months ago
gitignore	Git lab blog	6 months ago
gitlab-ci.yml	witness runner working	1 month ago
Dockerfile	Initial demo code	10 months ago

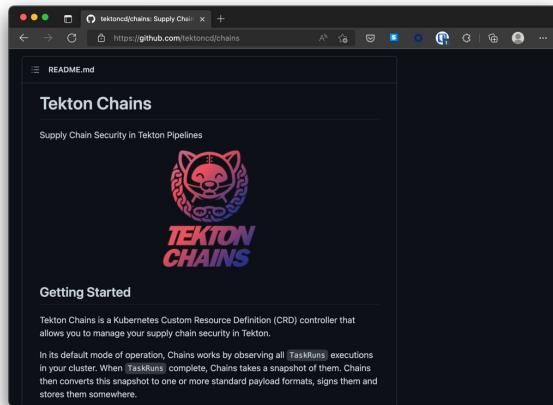
NDC { Sydney } @nielstanis

<https://gitlab.com/testifysec/demos/witness-demo>

<https://github.com/testifysec/witness>



Open Shift - Tekton - Tekton Chains



NDC { Sydney }

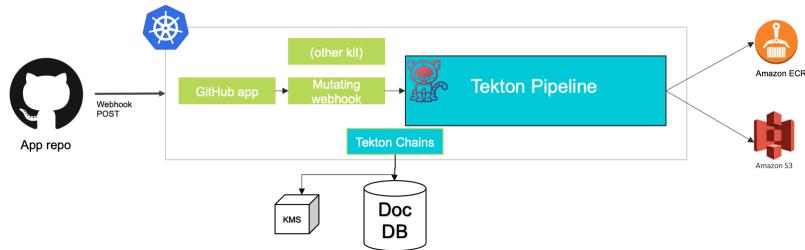
@nielstanis

<https://github.com/tektoncd/chains>

0101
0101

SolarWinds Project Trebuchet

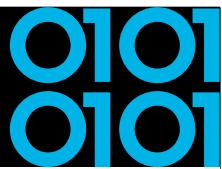
Pipeline With Attestations



NDC { Sydney }

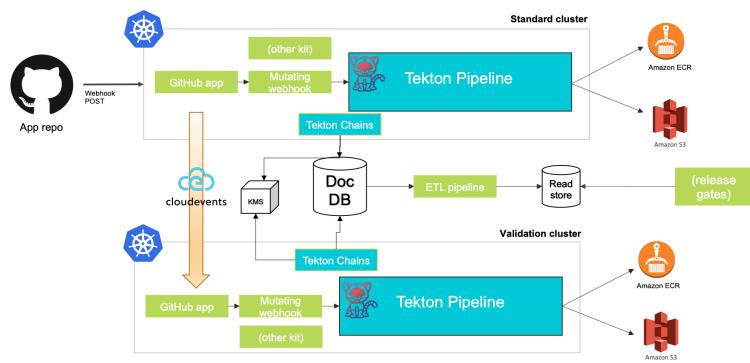
@nielstanis

https://static.sched.com/hosted_files/supplychainsecurityconna21/df/SupplyChainCon-TrevorRosen-Keynote.pdf
<https://www.youtube.com/watch?v=1-tMRxqMwTQ>



SolarWinds Project Trebuchet

Reading Results

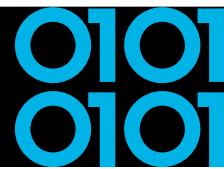


NDC { Sydney }

@nielstanis

https://static.sched.com/hosted_files/supplychainsecurityconna21/df/SupplyChainCon-TrevorRosen-Keynote.pdf

<https://www.youtube.com/watch?v=1-tMRxqMwTQ>



Grafeas and Kritis by Google

- Google released in 2019
- Grafeas - Component Metadata API
 - Container Analysis API on Google Cloud Platform
- Kritis - Deployment Authorization for Kubernetes Apps
 - Binary Authorization on Google Cloud Platform



NDC { Sydney }

@nielstanis

<https://grafeas.io/>

<https://github.com/grafeas/kritis/blob/master/docs/binary-authorization.md>

<https://www.infoq.com/presentations/supply-grafeas-kritis/>

<https://www.youtube.com/watch?v=hOzH3mOApjs>

<https://www.youtube.com/watch?v=05zN-YQxEAM>

Azure Policy

0101
0101

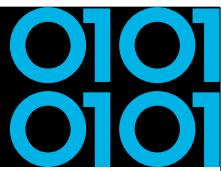
The screenshot shows a Microsoft Docs page for Azure Policy. The main title is "Tutorial: Create and manage policies to enforce compliance". Below it is a table of contents for "Create and manage Azure Policy", which includes sections like "Create a custom policy definition", "Manage tag governance", and "Route policy state change events". To the right of the table of contents, there's a sidebar with links for "Overview", "Quickstarts", and "Tutorials". The URL of the page is https://docs.microsoft.com/en-us/azure/governance/policy/tutorials/create-manage-policies-enforce-compliance.

NDC { Sydney }

@nielstanis

<https://devblogs.microsoft.com/devops/secure-software-supply-chain-with-azure-pipelines-artifact-policies/>

<https://docs.microsoft.com/en-us/azure/devops/pipelines/process/artifact-policy?view=azure-devops>

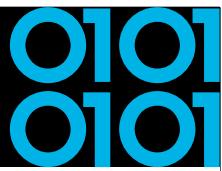


Conclusion

- 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.

NDC { Sydney }

@nielstanis



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!

NDC { Sydney }

@nielstanis

010101010101010101010101010101
01 VERACODE 0101010101010101010101
01010101010101010101010101010101

Thanks! Questions?

<https://github.com/nielstanis/ndcsydney2022>
ntanis at veracode.com
@nielstanis on Twitter

