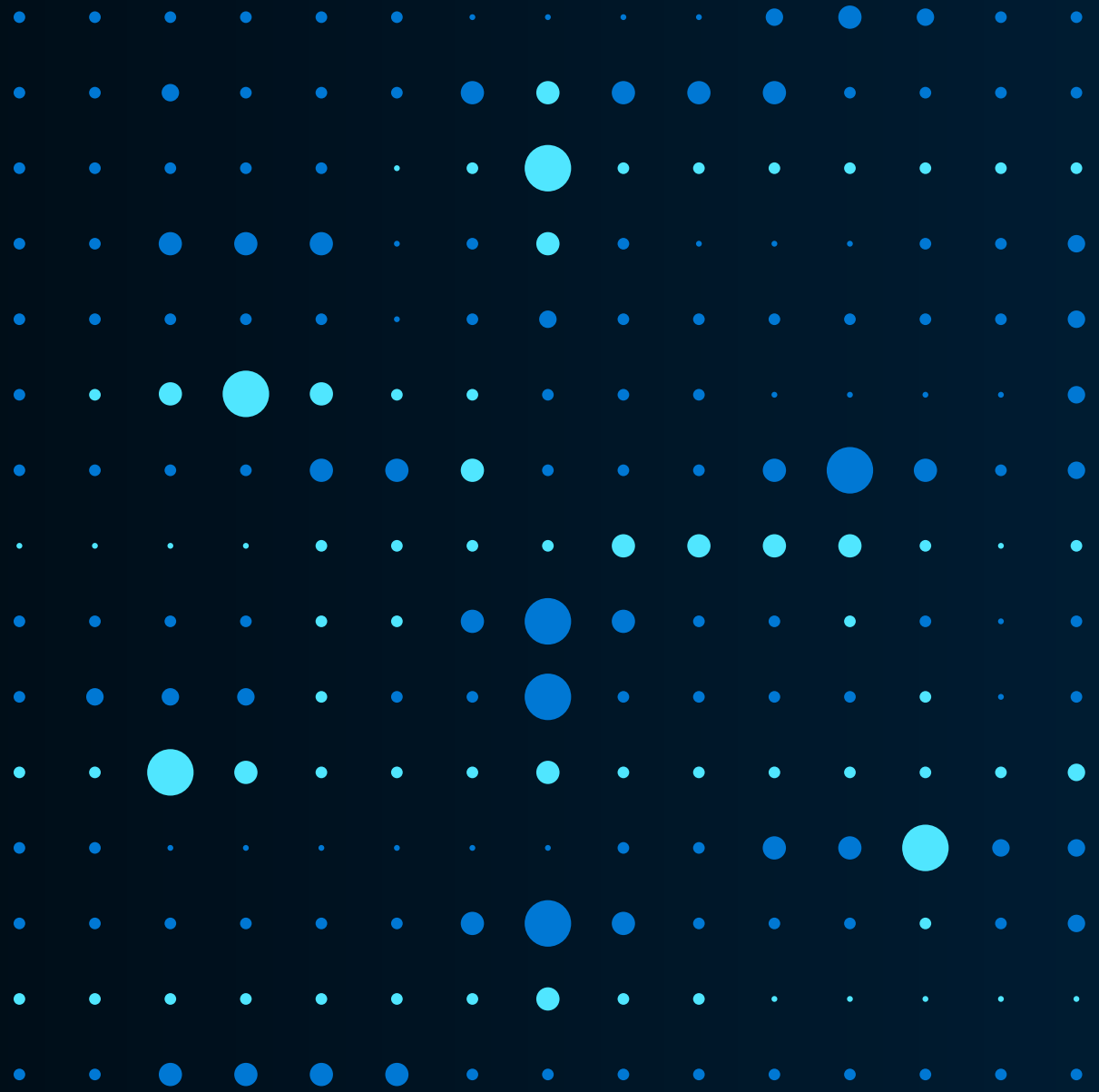
Microsoft

SBOM Lifecycle

Adrian Diglio

Principal PM Manager of Secure Software Supply Chain (S3C)

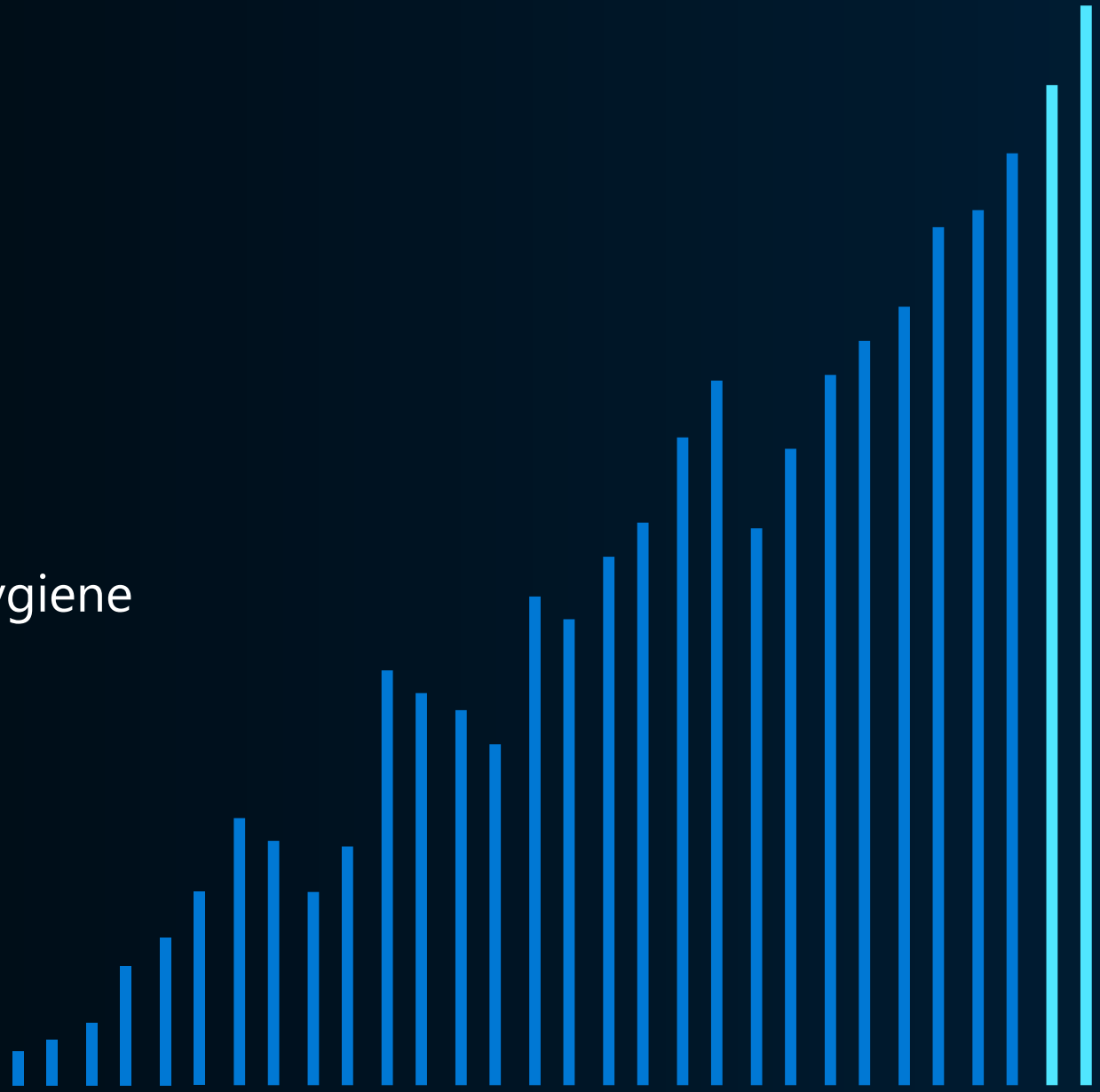
5/31/2023



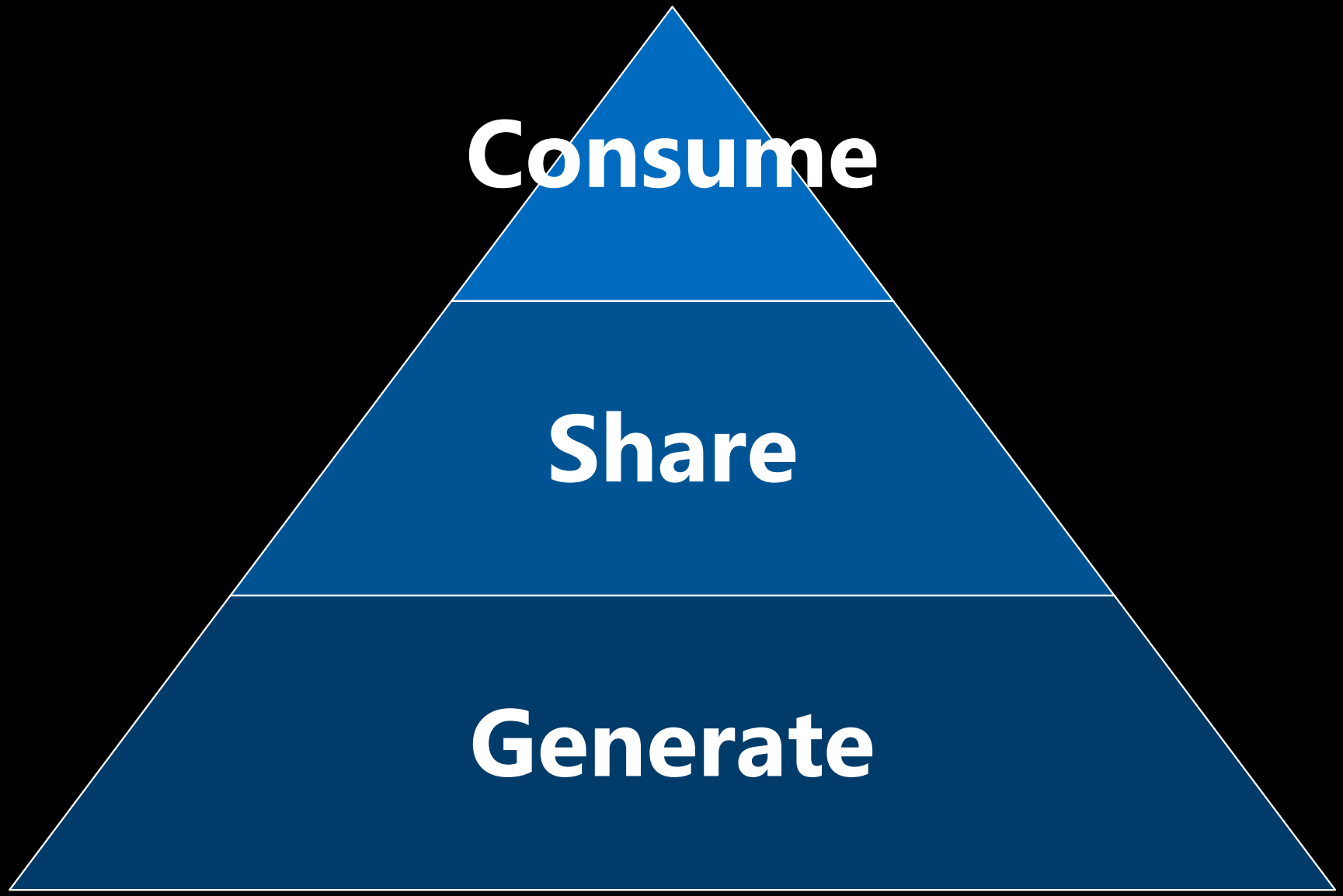
Microsoft

Agenda

- SBOM Pyramid
- Generating Different Types of SBOMs
- Sharing SBOMs based on software type
- Preparing for SBOM Consumption
- Security practices to help with SBOM hygiene



The SBOM Pyramid



CISA SBOM Types

SBOM Type	Definition
Design	SBOM of intended, planned software project or product with included components (some of which may not yet exist) for a new software artifact.
Source	SBOM created directly from the development environment, source files, and included dependencies used to build a product artifact.
Build	SBOM generated as part of the process of building the software to create a releasable artifact (e.g., executable or package) from data such as source files, dependencies, built components, build process ephemeral data, and other SBOMs.
Analyzed	SBOM generated through analysis of artifacts (e.g., executables, packages, containers, and virtual machine images) after its build. Such analysis generally requires a variety of heuristics. In some contexts, this may also be referred to as a "3rd party" SBOM
Deployed	SBOM provides an inventory of software that is present on a system. This may be an assembly of other SBOMs that combines analysis of configuration options, and examination of execution behavior in a (potentially simulated) deployment environment.
Runtime	SBOM generated through instrumenting the system running the software, to capture only components present in the system, as well as external call-outs or dynamically loaded components. In some contexts, this may also be referred to as an "Instrumented" or "Dynamic" SBOM.

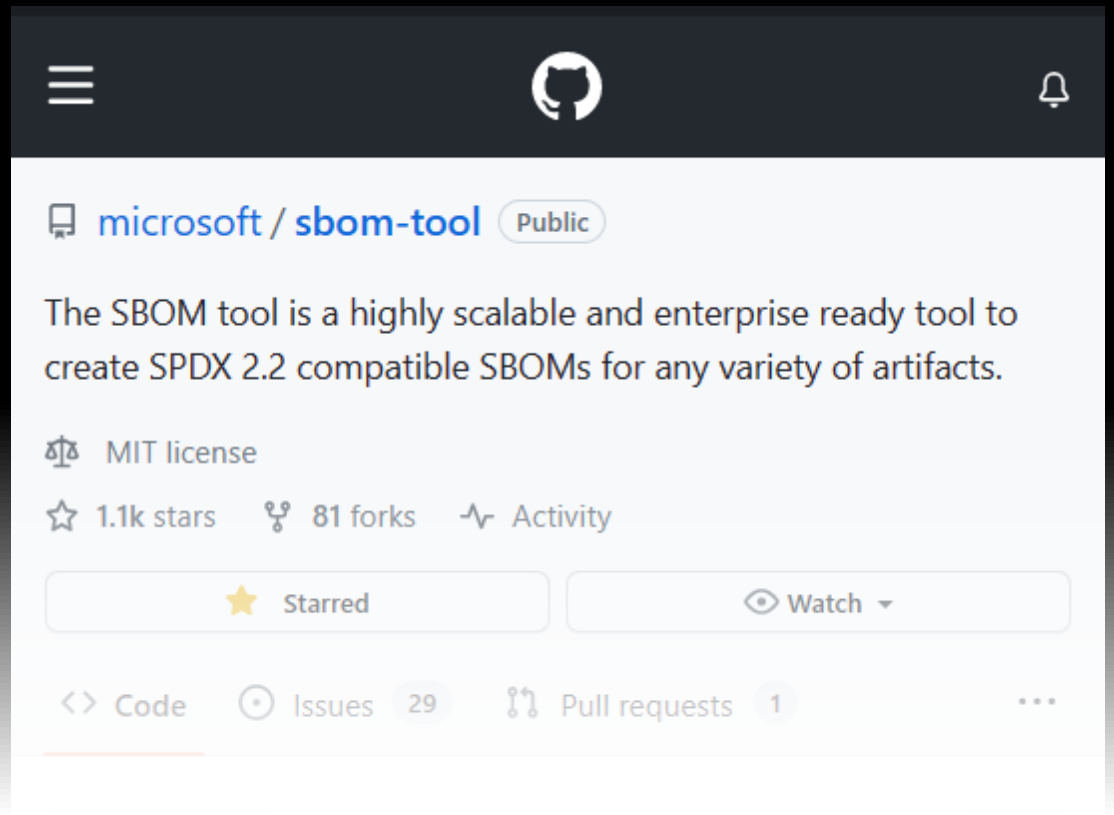
Microsoft's SBOM-tool
GitHub's Self-Service SBOM

Microsoft's Defender for
IoT Firmware Analysis

Microsoft's open source SBOM-Tool

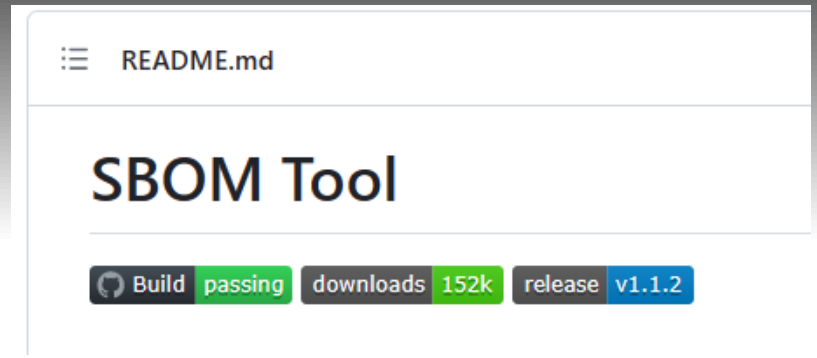
- Microsoft open sourced its SBOM generation tool

github.com/microsoft/sbom-tool



Features:

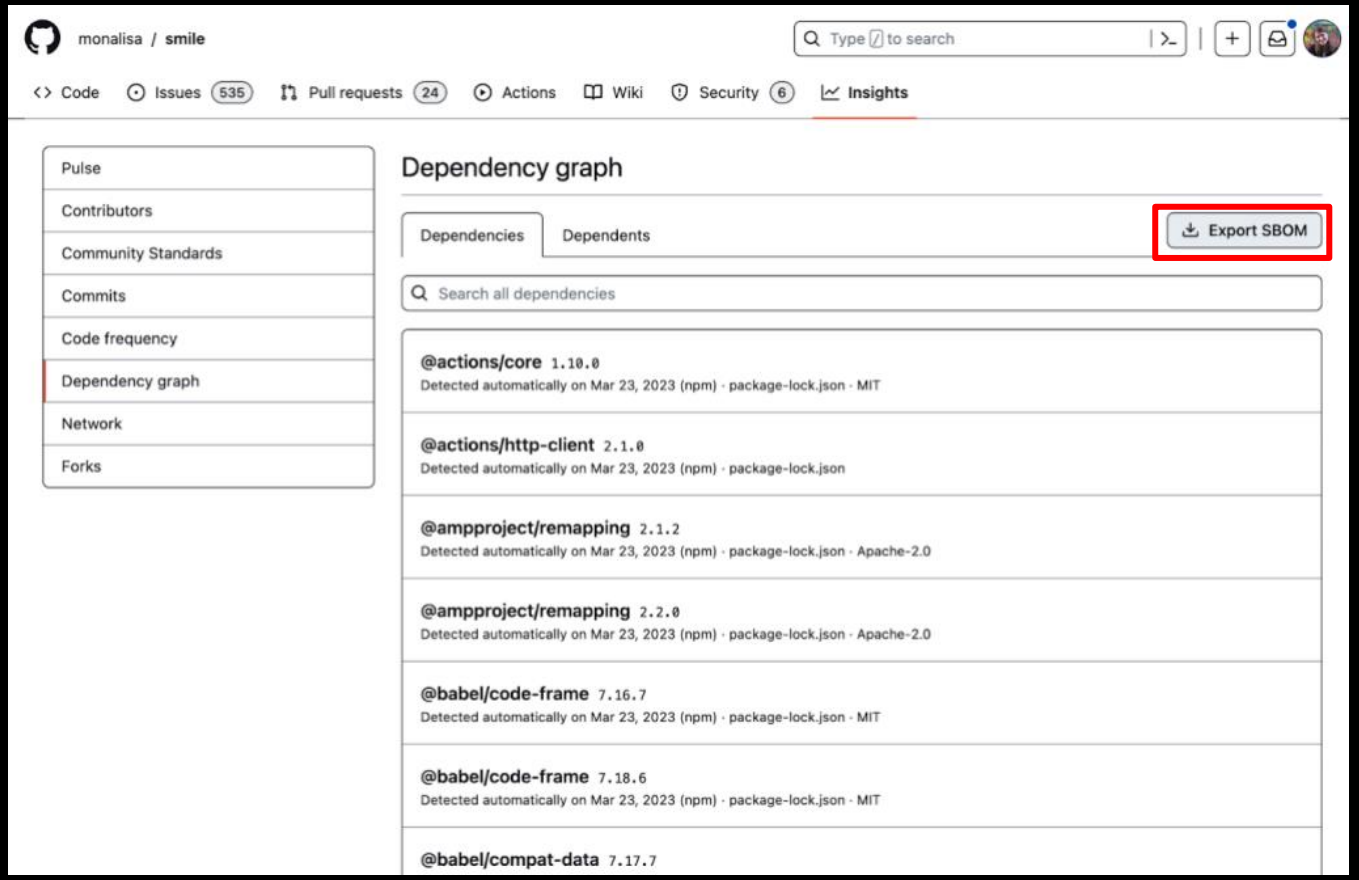
- Cross-platform support (Windows, Linux, and Mac)
- General purpose dependency detection (NPM, NuGet, Pypi, CocoaPods, Maven, Golang, Rust Crates, RubyGems, containers and their Linux packages via Syft, Gradle, Ivy, and vcpkg
- Able to detect transitive dependencies
- Produces NTIA-compliant SPDX SBOMs
- Ideally used for Build SBOMs, but can produce Source SBOMs as well
- Can detect and reference other SBOMs



GitHub's Self-Service SBOMs

[Introducing self-service SBOMs | The GitHub Blog]

- Ability to produce SBOMs On-Demand or programmatically generate them via the SBOM gh CLI extension
- REST API for generating an SBOM is coming soon
- Able to upload SBOM to dependency graph to receive Dependabot alerts

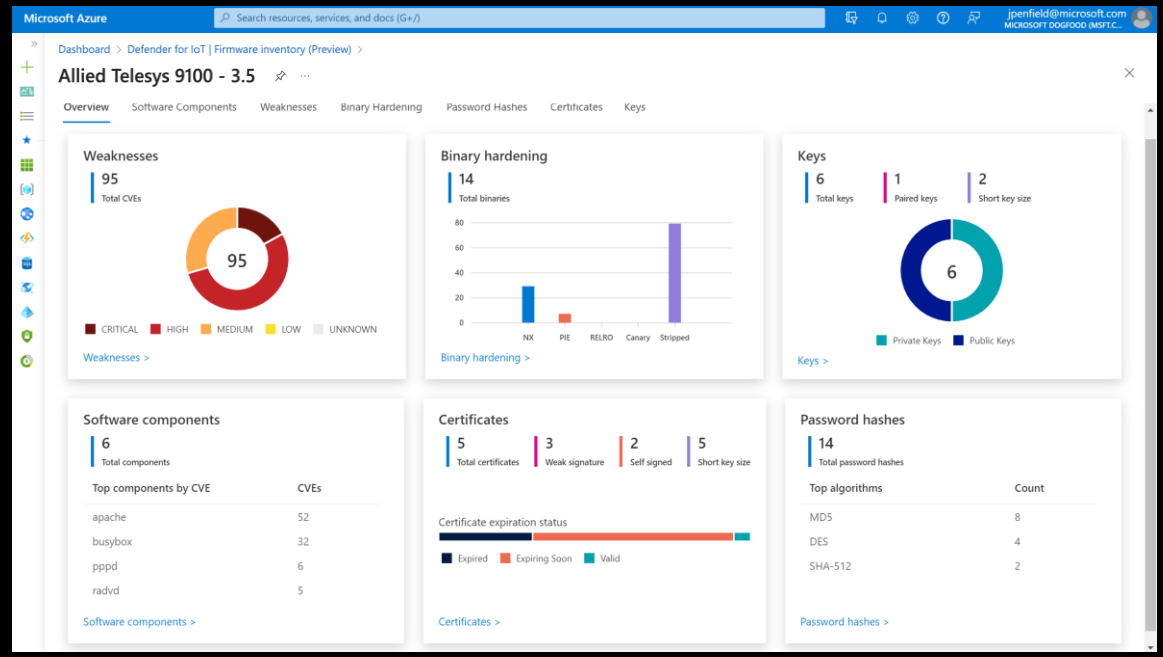


Private Preview

Defender for IoT Firmware Analysis

Automated identification of potential firmware security vulnerabilities

- SBOM
- Known vulnerabilities (CVE)
- Binary hardening
- Crypto Material
- Built-in accounts / weak passwords
- Continuous monitoring of new threats



Owner/operator

- Surfaced in Defender for IoT
- Brownfield device visibility
- Ongoing monitoring
- Enforce security policy before accepting delivery

OEM

- Part of Secure Development Lifecycle (SDL)
- Future GitHub integration
- Enforce security policy before ship
- Supply chain validation
- Regulatory (vuln mgmt / SBOM, etc.)

Share: SBOMs by Type

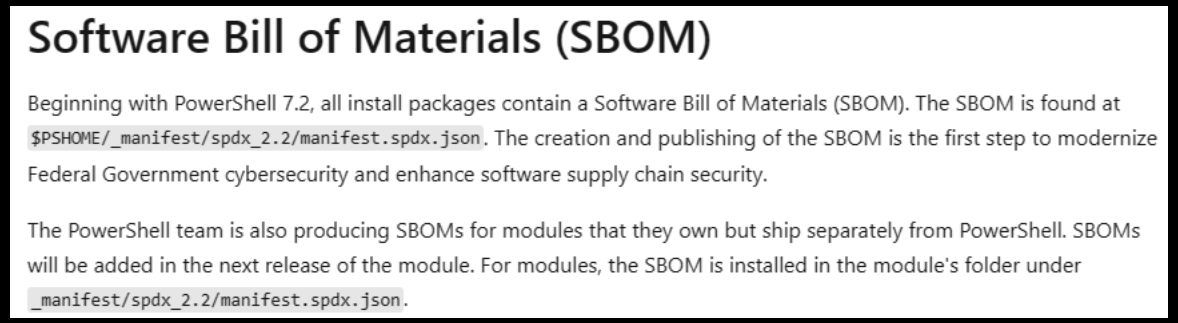
Software Type	Delivery Method	Example Links	Considerations
Operating System, Installable Binary	Included in compiled code	PowerShell security features - PowerShell | Microsoft Learn	Components worried about reproducibility might keep SBOMs detached (stored online)
Container image	Detached, stored side-by-side in the registry	Attach, push, and pull supply chain artifacts - Azure Container Registry | Microsoft Learn	
Virtual Hard Drive (VHD)	Detached (stored online)		
Embedded, IoT, Firmware	Detached (stored online)		If included in the compiled code, it couldn't be extracted
Open Source Components	Multiple (depends on ecosystem). Stored in repo, detached, or included in package	SBOMs at Anaconda	
Cloud Service	Detached (stored online)		
Microservices	Detached (stored online)		

SBOM Sharing

Given that so many software types would favor storing SBOMs detached from the software it represents, **there is a need to strongly link a software artifact or service to its SBOM stored online**

For the U.S. Presidential Executive Order 14028, in accordance with OMB M-22-18, Microsoft is first going to provide SBOMs to Federal Agencies when requested

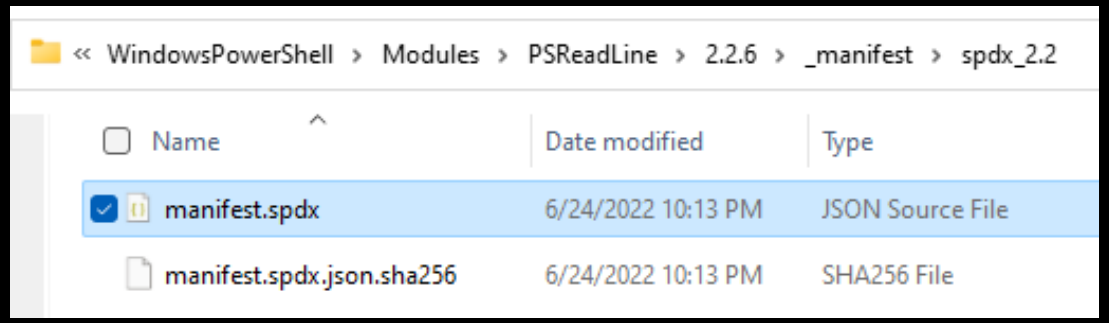
As we mature in our processes, individual product teams may choose to include SBOMs in their shipping bits – just as PowerShell did starting with version 7.2



Software Bill of Materials (SBOM)

Beginning with PowerShell 7.2, all install packages contain a Software Bill of Materials (SBOM). The SBOM is found at `$PSHOME/_manifest/spdx_2.2/manifest.spdx.json`. The creation and publishing of the SBOM is the first step to modernize Federal Government cybersecurity and enhance software supply chain security.

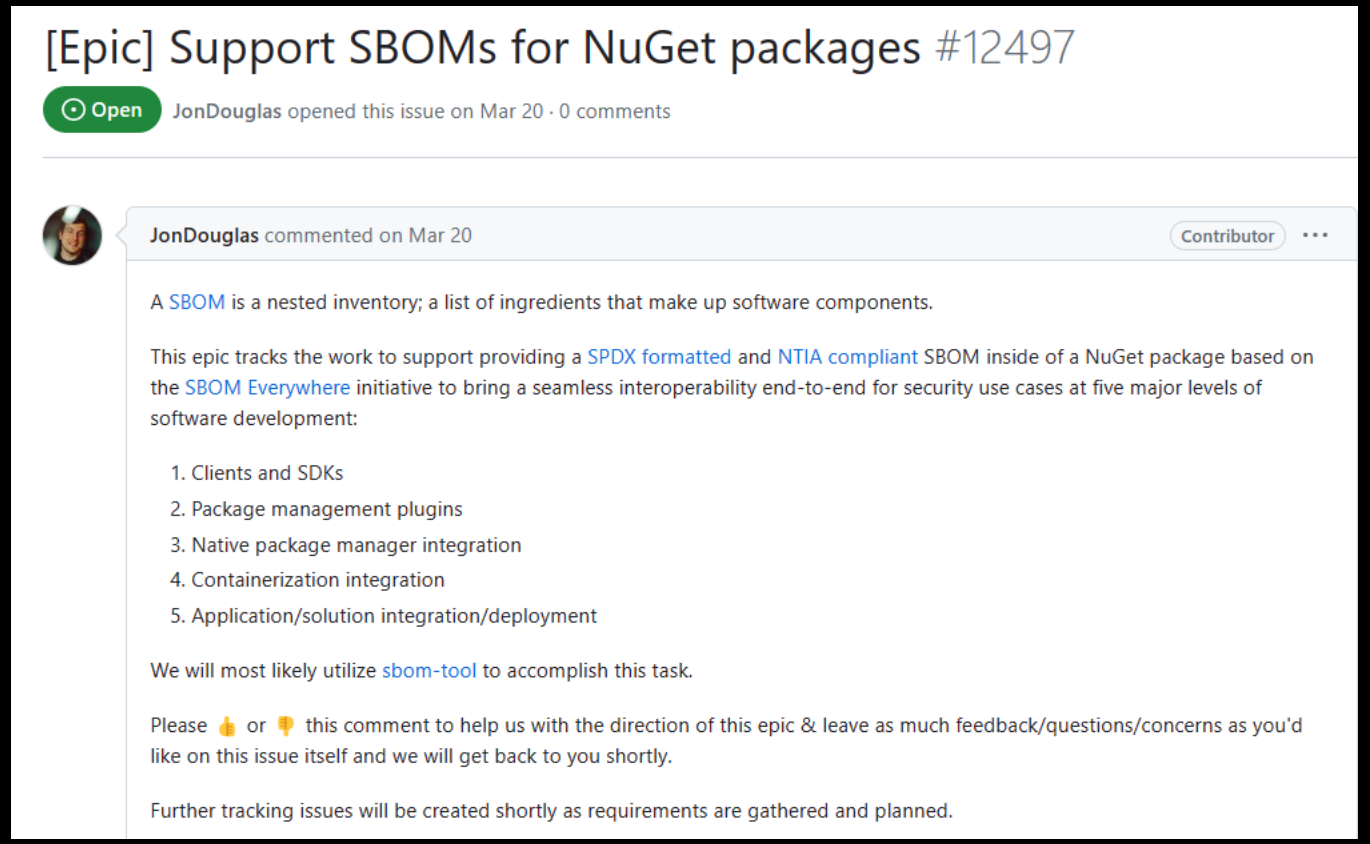
The PowerShell team is also producing SBOMs for modules that they own but ship separately from PowerShell. SBOMs will be added in the next release of the module. For modules, the SBOM is installed in the module's folder under `_manifest/spdx_2.2/manifest.spdx.json`.



Supporting the SBOM Everywhere Initiative

OpenSSF Mobilization Plan identified 10 workstreams, including SBOMs Everywhere

NuGet's 2023 Plan includes supporting SBOM Generation natively within Clients and SDKs and more



[Epic] Support SBOMs for NuGet packages #12497

Open JonDouglas opened this issue on Mar 20 · 0 comments

JonDouglas commented on Mar 20 Contributor

A SBOM is a nested inventory; a list of ingredients that make up software components.

This epic tracks the work to support providing a SPDX formatted and NTIA compliant SBOM inside of a NuGet package based on the SBOM Everywhere initiative to bring a seamless interoperability end-to-end for security use cases at five major levels of software development:

1. Clients and SDKs
2. Package management plugins
3. Native package manager integration
4. Containerization integration
5. Application/solution integration/deployment

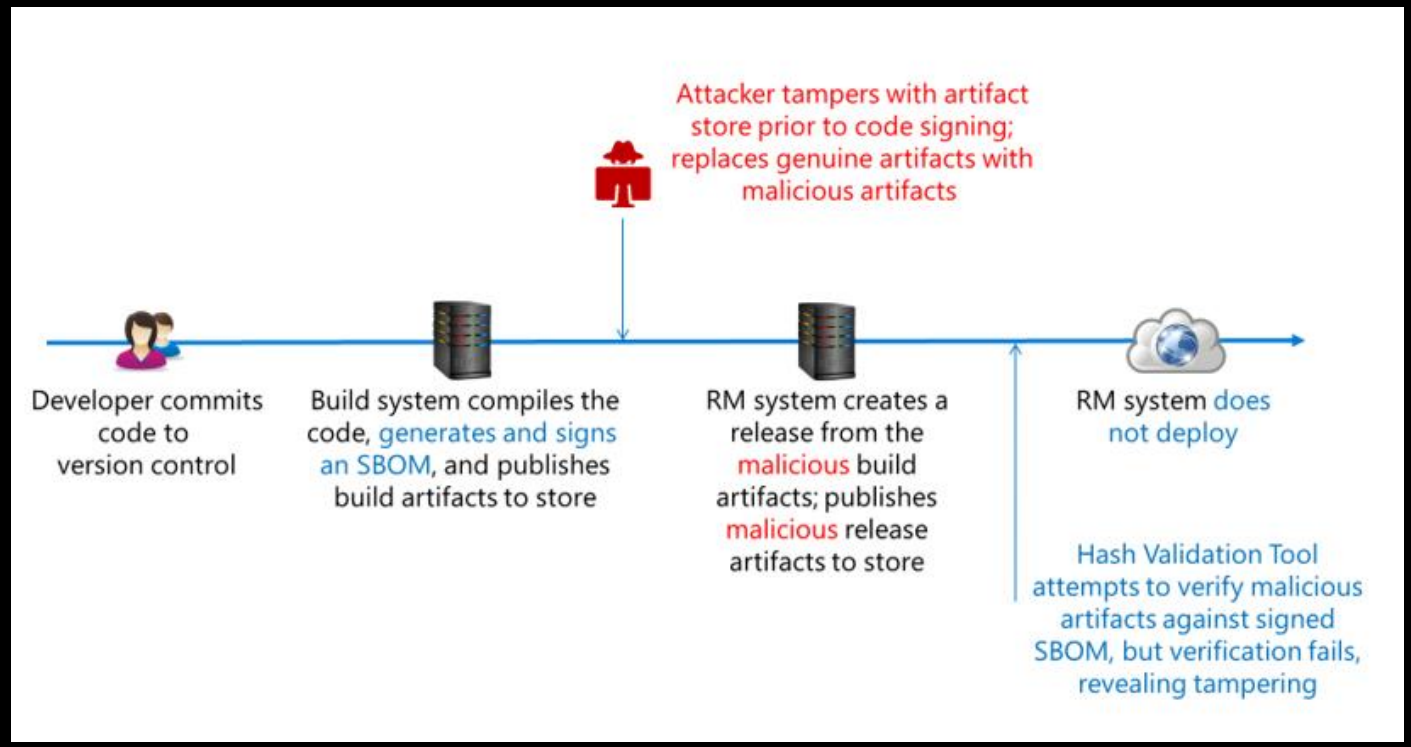
We will most likely utilize sbom-tool to accomplish this task.

Please 👍 or 👎 this comment to help us with the direction of this epic & leave as much feedback/questions/concerns as you'd like on this issue itself and we will get back to you shortly.

Further tracking issues will be created shortly as requirements are gathered and planned.

SBOM Validation at Release

- In a 28-day period, Microsoft generated over 1.59 million SBOMs at build time
  - Not all builds go through release
- Validating SBOMs at release ensures that what we built is what we released (integrity verification)
- Storing and sharing SBOMs from a release is the area of our focus today



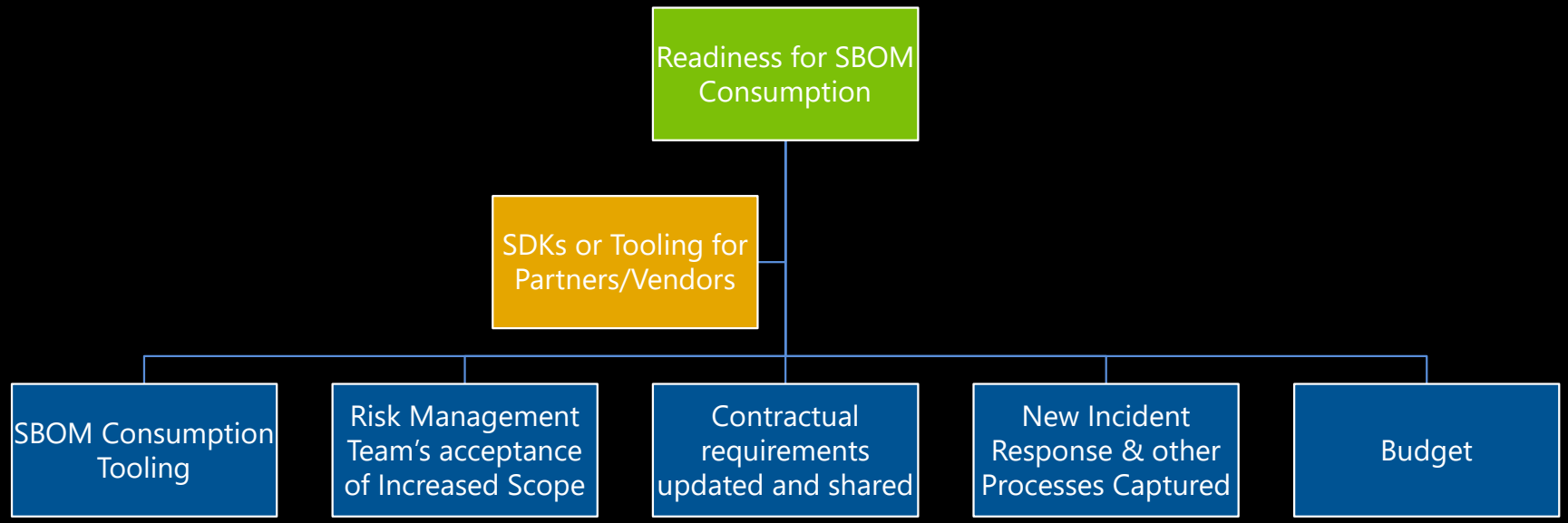
SBOM Sharing Lifecycle Phases

Source: CISA SBOM Sharing Lifecycle Report

As more Governments, contracts, and other enforcement mechanisms start requiring SBOMs, there will be a tipping point where SBOMs will soon become commonplace across the industry

The demand to discover and access SBOMs will likely increase as organizations, enterprises, and small and medium businesses adopt SBOM Consumption/Management tooling into their Risk Management practices

Organizational Preparation for SBOM Consumption



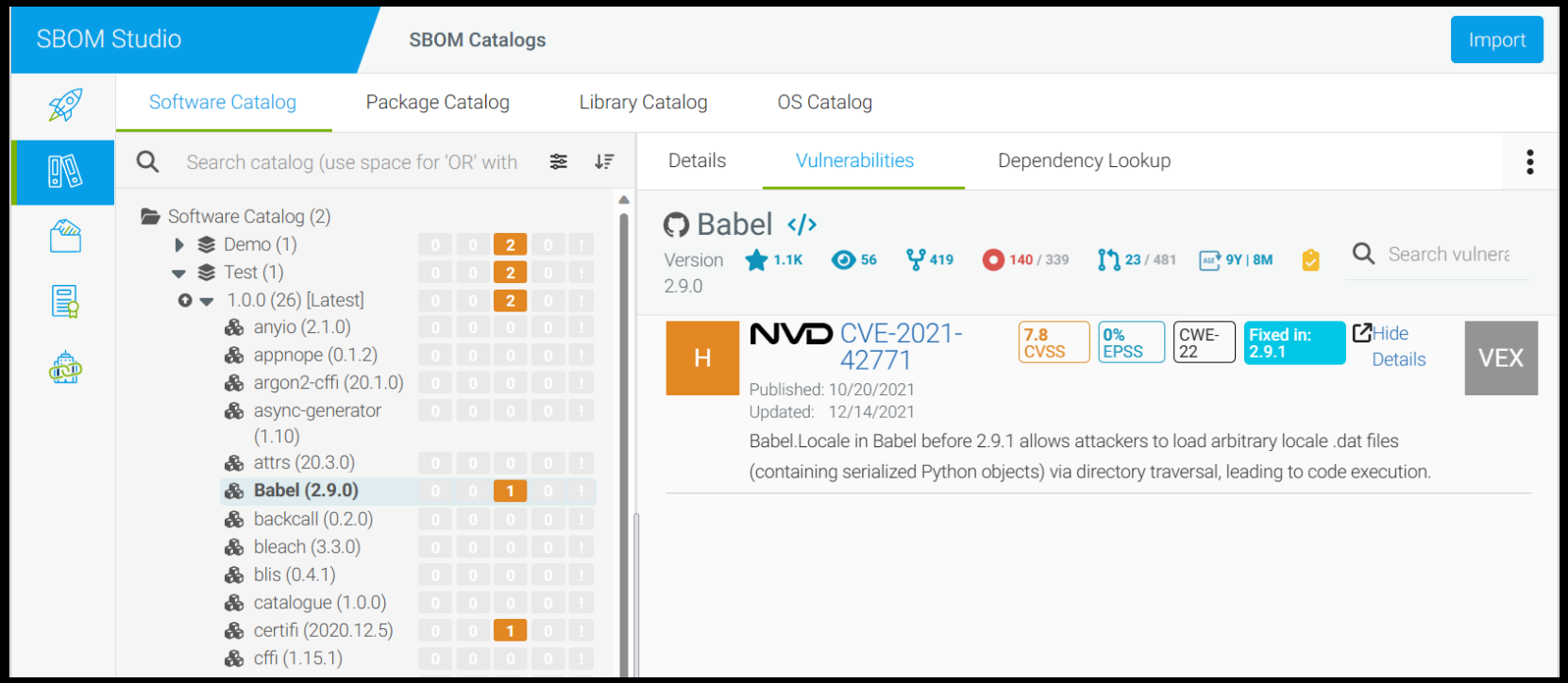
Example SBOM Consumption Tooling

Software consumers will load SBOMs into SBOM Management tools that will reveal:

1) OSS licenses
2) OSS vulnerabilities

An SBOM's core purpose is for incident response. When the next Log4J happens, our customers can answer the question "Am I affected?" and "What is affected?"

Many Organizations will have SBOMs available before they have VEX statements available



*This is a 3P tool that represents what software consumers will see

Hygiene for Producing Better SBOMs

OpenSSF Secure Supply Chain Consumption Framework (S2C2F)

- The S2C2F is a set of practices and requirements, organized into a maturity model, on how to securely consume OSS into the developer's DevOps workflow
  - This is referring to NuGet, NPM, Maven, Pypi, Rust Crates, Rubygems, Golang, vcpkg, etc.
- The S2C2F pairs with producer-focused frameworks such as NIST, NSA ESF, SLSA, CIS, and more

Real-world OSS supply chain threats

Threats	Real examples	Mitigation via S2C2F	Framework requirement reference
Accidental vulnerabilities in OSS code or containers that we inherit	SaltStack	Automated patching, display OSS vulnerabilities as pull requests	UPD-2, UPD-3
Intentional vulnerabilities/backdoors added to an OSS code base	phpMyAdmin	Perform proactive security review of OSS	SCA-5
A malicious actor compromises a known good OSS component and adds malicious code into the repo	ESLint incident	Ability to block ingestion via malware scan, curated feed, all packages are scanned for malware prior to download	ING-3, ENF-2, SCA-4
A malicious actor creates a malicious package that is similar in name to a popular OSS component to trick developers into downloading it	Typosquatting	OSS provenance analysis, curated feed, all packages are scanned for malware prior to download	AUD-1, ENF-2, SCA-4
A malicious actor compromises the compiler used by the OSS during build, adding backdoors	CCleaner	Rebuilding OSS on trusted build infrastructure ensures that packages don't have anything injected at build time	REB-1
Dependency confusion, package substitution attacks	Dependency Confusion	Securely configure your package source mapping, curated feed	ENF-1, ENF-2
An OSS component adds new dependencies that are malicious	Event-Stream incident	All packages are scanned for malware prior to download, curated feed	SCA-4, ENF-2
The integrity of an OSS package is tampered after build, but before consumption	How to tamper with Electron apps	Digital signature or hash verification, SBOM validation	AUD-3, AUD-4
Upstream source can be removed or taken down which can then break builds that depend on that OSS component or container	left-pad	Use package-caching solutions, mirror a copy of OSS source code to an internal location for Business Continuity and Disaster Recovery (BCDR) scenarios	ING-2, ING-4
OSS components reach end-of-support/end-of-life and therefore don't patch vulnerabilities	log4net CVE-2018-1285	Scan OSS to determine if it is at end-of-life	SCA-3
Vulnerability not fixed by upstream maintainer in desired timeframe	Prototype Pollution in lodash	Implement a change in the code to address a zero-day vulnerability, rebuild, deploy to your organization, and confidentially contribute the fix to the upstream maintainer.	FIX-1
Bad actor compromises a package manager account (e.g. npm), with no change to the corresponding open source repo, and uploads a new malicious version of a package	Ua-parser-js	OSS provenance analysis, curated feed, scan OSS for malware	AUD-1, ENF-2, SCA-4

Secure Supply Chain Consumption Framework Maturity Model

Level 1	Level 2	Level 3	Level 4
Minimum OSS Governance Program	**Secure Consumption and Improved MTTR**	**Malware Defense and Zero-Day Detection**	**Advanced Threat Defense**
• Use package managers • Local copy of artifact • Scan with known vulns • Scan for software licenses • Inventory OSS • Manual OSS updates	• Scan for end life • Have an incident response plan • Auto OSS updates • Alert on vulns at PR time • Audit that consumption is through the approved ingestion method • Validate integrity of OSS • Secure package source file configuration	• Deny list capability • Clone OSS source • Scan for malware • Proactive security reviews • Enforce OSS provenance • Enforce consumption from curated feed	• Validate the SBOMs of OSS consumed • Rebuild OSS on trusted infrastructure • Digitally sign rebuilt OSS • Generate SBOM for rebuilt OSS • Digitally sign protected SBOMs • Implement fixes

The framework lists out the requirements and organizes it into a maturity model, where each level has different themes.

S2C2F is referenced in the forthcoming NSA ESF guidebook

Checkout the S2C2F Guide today: [s2c2f/framework.md at main · ossf/s2c2f · GitHub]

Questions?

Adrian Diglio

- LinkedIn: Adrian Diglio | LinkedIn
- Email: Adrian.Diglio@Microsoft.com

Further Reading

- Generating Software Bills of Materials (SBOMs) with SPDX at Microsoft - Engineering@Microsoft
- Microsoft contributes S2C2F to OpenSSF | Microsoft Security Blog
- The Journey to Secure the Software Supply Chain at Microsoft - Engineering@Microsoft