Enduring Security Framework

18 SEPTEMBER 2024

CAROL LEE
NSA – CHIEF OF THE CENTER FOR ASSURED SOFTWARE
CALEE3@UWE.NSA.GOV

EO 14028 & the Enduring Security Framework (ESF) SOLARPANEL

Current Landscape:

- Proliferation of Open Source Software (OSS) components; many with vulnerabilities
  - Attacks to the supply chain more prevalent in the last 5 years (ex. Log4j and SolarWinds)
  - SolarWinds Software Suite attack and Log4j exploit

Response:

- The ESF Software Supply Chain Working Panel was established
  - Assess the current state of the security of software supply chain processes & identify solutions to risks
- Presidential Executive Order EO 14028 *Improving the Nation's Cybersecurity*[1] (May 2021)
  - Focuses on the security and integrity of the software supply chain and emphasizes the importance of secure software development environments
- Software Bill of Materials (SBOM) support checking licenses and vulnerable versions
- Vulnerability Exploitability eXchange (VEX)[2-3]

EO 14028 & the Enduring Security Framework (ESF) SOLARPANEL

- ESF working panel developed guidance and best practices for software acquisition, development, distribution, and management
  - Guidance used by NIST in support of SP800-161 Cybersecurity Supply Chain Risk Management (C-SCRM) and their milestones for EO14028
  - The National Security Council
  - White House's Open Source Software (OSS) Summit
- Leveraged the best practices and complementary efforts (ex. NIST SP 800-218 SSDF, SLSA)
- Produced five *Securing the Software Supply Chain* Recommended Practices Guides[4]

NIST – National Institute of Standards and Technology
SSDF – Secure Software Development Framework
SLSA – Supply-chain Levels for Software Artifacts framework

ESF Software Supply Chain - Threats & Best Practices

Phase I

High level guidance and recommended practices for enhancing security and resiliency in software supply chain processes

- Customers – Basis for a risk management approach to acquisitions, deployment, & maintenance of software products
- Suppliers – Product security through updates addressing emergent threats & identified code weaknesses
- Developers – Secure software assurance through utilizing practices, principles & testing throughout the SDLC

Securing the Software Supply Chain: Recommended Practices Guide

1. Customers
   - Procurement/Acquisition
   - Deployment
   - Operations

2. Suppliers
   - Prepare the Organization
   - Protect Software
   - Produce Well-Secured Software
   - Respond to Vulnerabilities

3. Developers
   - Secure Product Criteria and Management
   - Develop Secure Code
   - Verify 3rd Party Components
   - Harden the Build Environment
   - Code Delivery

4. Appendices
   - Crosswalk between Uses Cases & NIST SSDF
   - Dependencies between the 3 groups
   - SLSA Levels
   - Artifacts and Checklist
   - References & Acronyms

Securing the Software Supply Chain: Recommended Practices Guide

1. Customers
   - Procurement/Acquisition
   - Deployment
   - Operations
2. Suppliers
   - Prepare the Organization
   - Protect Software
   - Produce Well-Secured Software
   - Respond to Vulnerabilities
3. Developers
   - Secure Product Criteria and Management
   - Develop Secure Code
   - Verify 3rd Party Components
   - Harden the Build Environment
   - Code Delivery
4. Appendices
   - Crosswalk between Uses Cases & NIST SSDF
   - Dependencies between the 3 groups
   - SLSA Levels
   - Artifacts and Checklist
   - References & Acronyms

Customers

Verifying the Supply Chain across acquisition, deployment and operational phases

- Examples Threats:
  - Security requirements are not specific to the domain or mission & there are gaps
  - Products have unknown features or components that may have vulnerabilities
- Example Mitigations:
  - Select security requirements specific to program
  - Create a relationship with the vendor to
    - ➢ Understand the security aspects in creation, delivery and updates
    - ➢ Receive notifications of cyber incidents & changes in ownership and the resulting risk security impacts
  - Require attestation
  - Check the integrity of the accepted delivery or update
  - Review SBOM & Test product
  - Implement continuous monitoring

ENDURING SECURITY FRAMEWORK

SECURING THE SOFTWARE SUPPLY CHAIN

CUSTOMERS

User Organizations (i.e. the Customer) follow a series of key phases in the acquisition, deployment, and operation of software products.

Customer teams specify to and rely on vendors for providing key artifacts (e.g. SBOM) and mechanisms to verify the software product, its security properties, and attest to the SDLC security processes and procedures.

CUSTOMERS · Procurement Acquisition • Deployment • Operations
SECURE INFRASTRUCTURE · Development Process
DEVELOPERS · Build Engineer • S/W Engineer • QA Engineer
SUPPLIERS · Producer • ISP Producer • Services • Product
Support
Support

Processes Involved in the Key Phases

- Requirements Definition
- Supply Chain Risk Management
- Product Evaluation and Selection
- Contracting Deployment/Integration
- IT Operations
- Security Operations

Post procurement, the customer evaluates the product, plans for integration and roll out, and then performs necessary functions and steps for deploying the software into the environment.

Part of the roll out process includes coordination with security operations, oversight boards, and the internal user community. Deployment processes incorporate procedures for product updates, upgrades, and End of Life (EOL) or removal. Finally, the customer implements procedures, processes, controls, and policies on the use and operations of the software product.

THREATS

- Product (or product update/upgrade)
  - Undocumented features, malicious or risky code/functionality, vulnerable code or components
  - Changes in functionality or security assumptions between evaluation and deployment
- Vendor
  - Change in ownership can impact risk assessment of product (control, geo location, ownership, security incidents)
  - Poor enterprise and/or development security hygiene

RECOMMENDED BEST PRACTICES

- Require and verify attestation (e.g. SBOM) for the product
- Perform Security, Environmental, and Functionality tests on the software product
- Check integrity mechanisms (e.g., product file hashes, config file hashes) during deployment
- Require notification from supplier
  - Changes of ownership/control, geo location, 3rd parties to software product)
  - Cyber incidents and investigations and
  - Mitigations/impacts to product or development environment of the product at time of delivery
- Require self-attestation of cybersecurity hygiene of the vendor/supplier development process and the infrastructure supporting the development of their product to inform the risk management determination by the user organization
- Continuous Security Monitoring of product as part of Security Operations

ENDURING SECURITY FRAMEWORK | NSAESF@CYBER.NSA.GOV | CYBERCENTER.NSA.GOV

Securing the Software Supply Chain: Recommended Practices Guide

1. Customers
   - Procurement/Acquisition
   - Deployment
   - Operations

2. Suppliers
   - Prepare the Organization
   - Protect Software
   - Produce Well-Secured Software
   - Respond to Vulnerabilities

3. Developers
   - Secure Product Criteria and Management
   - Develop Secure Code
   - Verify 3rd Party Components
   - Harden the Build Environment
   - Code Delivery

4. Appendices
   - Crosswalk between Uses Cases & NIST SSDF
   - Dependencies between the 3 groups
   - SLSA Levels
   - Artifacts and Checklist
   - References & Acronyms

Suppliers

Ensuring Supply Chain integrity & security of software from production through delivery

- Examples Threats:
  - Attacks on the product archive or the delivery process
  - Inclusion of vulnerable 3rd party components
- Example Mitigations:
  - Define criteria for security checks (policies & SDLC)
  - Design software to meet security requirements
  - Configure software to have secure settings by default
  - Archive and protect software releases
  - Utilize mechanisms for verifying software release integrity
  - Establish and implement secure processes to monitor for issues and receive issues from the customer & deliver updates and patches

Securing the Software Supply Chain: Recommended Practices Guide

1. Customers
   - Procurement/Acquisition
   - Deployment
   - Operations

2. Suppliers
   - Prepare the Organization
   - Protect Software
   - Produce Well-Secured Software
   - Respond to Vulnerabilities

3. Developers
   - Secure Product Criteria and Management
   - Develop Secure Code
   - Verify 3rd Party Components
   - Harden the Build Environment
   - Code Delivery

4. Appendices
   - Crosswalk between Uses Cases & NIST SSDF
   - Dependencies between the 3 groups
   - SLSA Levels
   - Artifacts and Checklist
   - References & Acronyms

Developers

Securely developing and protecting the product within the development & build environment & delivery process

- Examples Threats:
  - Intentional injection of malicious or accidental development of vulnerable code within a product
  - Incorporation of vulnerable 3rd-party source code or binary within a product
  - Exploitation of weaknesses within the build process used to inject malicious software within a component of a product
  - Modification of a product deliverable within the delivery mechanism, resulting in the injection of malicious software into the delivery to the customer
- Example Mitigations:
  - Take training & use secure software development practices
  - Use memory safe languages
  - Incorporate secure architecture and design & threat modeling
  - Instantiate a secure repository for 3rd party components & monitor
  - Harden the build and delivery environment (ex. reproducible builds)
  - Document security test plans, tests, fixes & results w/CVSS scores
  - Support release criteria (ex. no unacceptable vulnerabilities, SBOM)
  - Provide product support & vulnerability handling (ex. Incident response)
  - Sign packages and provide attestation

Securing the Software Supply Chain: Recommended Practices Guide

1. Customers
   - Procurement/Acquisition
   - Deployment
   - Operations

2. Suppliers
   - Prepare the Organization
   - Protect Software
   - Produce Well-Secured Software
   - Respond to Vulnerabilities

3. Developers
   - Secure Product Criteria and Management
   - Develop Secure Code
   - Verify 3rd Party Components
   - Harden the Build Environment
   - Code Delivery

4. Appendices
   - Crosswalk between Uses Cases & NIST SSDF
   - Dependencies between the 3 groups
   - SLSA Levels
   - Artifacts and Checklist
   - References & Acronyms

Attestation & Checklist Examples

Artifact Examples	Description/Purpose
High-level Secure Development Lifecycle Process document	Attestation to secure development practices wl • Secure Software Architecture/Design p • Attack surface investigation and threa • Secure Software Development/Progra • Software Security testing process • Source Control Check-In Process • Trusted Repository for Modules and F • Continuous Integration and Delivery (• Defect/Vulnerability Reporting and Cu • Code Reviews Process for Security Improvement • Continuous Verification of third-Party • Open-Source Management Practices • Hardening the Build Environment • Secure Relationship with a third-Party • Process to Secure the Signing Server • Final Package Validation process
Product Readiness checklist	Attestation to product release and secure shipp shipment which can cover: • No pending known critical bugs and v • Cryptographically signed components • Proper software licensing

Measurable Outcome/ Description	Practice Observed Yes, No, NA, Inc	SSDF Tasks	Artifact Examples	Document References
Secure Product Criteria & Management				
Do you define policies that specify risk-based software architecture and design requirements?		**PO.1.2**	Architecture/Design Documents	4.1 Secure Product Criteria and Management
Do you require team members to regularly participate in secure software architecture, design, development, and testing training and monitor their training completion?		**PO.2.2** **RV.3.4**	Training Completion Data/Statistics Developer Training Certificates	4.1 Secure Product Criteria and Management 4.2.1 Secure Software Development/ Programming Training
Do you have up to date threat models for all critical components your team ships that have been reviewed by a person trained in software security and do you make this document available to other teams that pick up your component?		**PW.1.1, PW.2.1**	Threat Model Results Document	2.3.3 Security/SCRM Operations 3.3.5 Test Executable Code 4.1 Secure Product Criteria and Management
Has your team held a black-box investigation for security?			Black box test results	4.1 Secure Product Criteria and Management 4.3.1 Third-Party Binaries Threat Scenario

Customer, Supplier & Developer Security Activities

Developer Security Activities

- Architecture & Design Review
- SW Threat Modeling
- Attack Surface Analysis
- Coding Standards
- Secure Software Development
- Secure Library Checks (Composition Analysis)
- Code & Executable Testing
- Secure Build & Delivery

Supplier Security Activities

- Security Deliver Software
- Protect Code Pipeline
- Ensure Security Requirements are Met
- Verify 3rd Party Software
- Test Executable Code
- Configure Secure Defaults
- Notify Customers of Vulnerabilities
- Respond to Vulnerabilities

Customer Security Activities

- Security Requirements Definition
- SCRM Requirements Definition
- Security Acceptance Testing
- SCRM & Product Integrity Validation
- Implement Risk & Security Controls for Product Deployment
- Product Lifecycle Management Security & SCRM Controls & Processes

Software Supply Chain - Threats & Best Practices

Phase II – Open Source Software (OSS) & Software Bill of Materials (SBOM)

- Built upon Phase I guidance
- "How to" guide for the Phase I activities
- Collaborate with existing threat models to better address attacks to OSS-related vulnerabilities
- Partner with other forums on OSS-related vulnerabilities to contribute to more widely acceptable guidance

Securing the Software Supply Chain: Recommended Practices Guide

1. SBOM Consumption
   - Component Information
   - Ingestion & Management for Enterprise
   - Mapping and Asset Management
2. Managing OSS & SBOMs
   - Evaluate OSS acceptance/deployment process
   - Protection of existing environment
   - Management of Identity and Access control
   - Update and distribution of SBOMs
   - Verification of package contents

Securing the Software Supply Chain: Recommended Practices Guide

1. SBOM Consumption
   - Component Information
   - Ingestion & Management for Enterprise
   - Mapping and Asset Management
2. Managing OSS & SBOMs
   - Prepare the Organization
   - Protect Software
   - Produce Well-Secured Software
   - Respond to Vulnerabilities

SBOM Consumption

Verifying the Supply Chain integrity across acquisition, deployment and operational phases

- Examples Threats:
  - Undocumented features, vulnerable code or components
  - Changes in ownership (control, geo-location, etc.)
  - Poor security hygiene
- Example Mitigations:
  - Verify attestation (e.g. SBOM) for the product
  - Require notification from supplier
    - Changes of ownership/control, geo location, 3rd parties to software product, etc.
    - Cyber incidents and investigations
    - Mitigations/impacts to product or development environment of the product at time of delivery
  - Require self-attestation of cybersecurity hygiene of vendor/supplier development process and infrastructure supporting development of the product
  - Implement continuous security monitoring as part of the Security Operations

ENDURING SECURITY FRAMEWORK

SECURING THE SOFTWARE SUPPLY CHAIN RECOMMENDED PRACTICES FOR

Software Bill of Materials Consumption

Software Bill of Material (SBOM) work has advanced since 2018 as a collaborative community effort, driven by National Telecommunications and Information Administration's (NTIA) multi-stakeholder process. An SBOM conveys information about what is in the software. Consumption of an SBOM received by a wide variety of customer organizations has emerged as a key building block in software security and software supply chain risk management. SBOMs may be correlated with threat feeds and other data to augment the value and scope of the content provided. The mere act of knowing that a supplier can provide a quality SBOM offers benefits to the software user/customer, since it offers a certain level of confidence that the software supplier is more likely to be able to respond to supply chain concerns. Full leverage of the power of SBOM requires the capabilities to turn the SBOM data into security intelligence, which can then drive security actions.

PROCESSES INVOLVED

- SBOM Formats
- Operationalize and Scale the use of a SBOM
- Baseline Component Information
- SBOM Delivery
- SBOM Acceptance/Validation
- Ingestion and Management for Enterprise
- Extraction, Transformation and Loading of SBOMsMapping and Asset Management

THREATS

- Product (or product update/upgrade)
  - Undocumented features, malicious or risky code/functionality, vulnerable code or components
  - Changes in functionality or security assumptions between evaluation and deployment
- Vendor
  - Change in ownership can impact risk assessment of product (control, geo location, ownership, security incidents)
  - Poor enterprise and/or development security hygiene

RECOMMENDED PRACTICES

- Require and verify attestation (e.g. SBOM) for the product
- Perform Security, Environmental, and Functionality tests on the software product
- Check integrity mechanisms (e.g., product file hashes, config file hashes) during deployment.
- Require notification from supplier
  - Changes of ownership/control, geo location, 3rd parties to software product)
  - Cyber incidents and investigations and
  - Mitigations/impacts to product or development environment of the product at time of delivery
- Require self-attestation of cybersecurity hygiene of the vendor/supplier development process and the infrastructure supporting the development of their product to inform the risk management determination by the user organization.
- Continuous Security Monitoring of product as part of Security Operations

NSAESF@CYBER.NSA.GOV
CYBERCENTER.NSA.GOV

SBOM Consumption – Process & Risk

Operationalizing an SBOM across verification, acquisition, risk & deployment

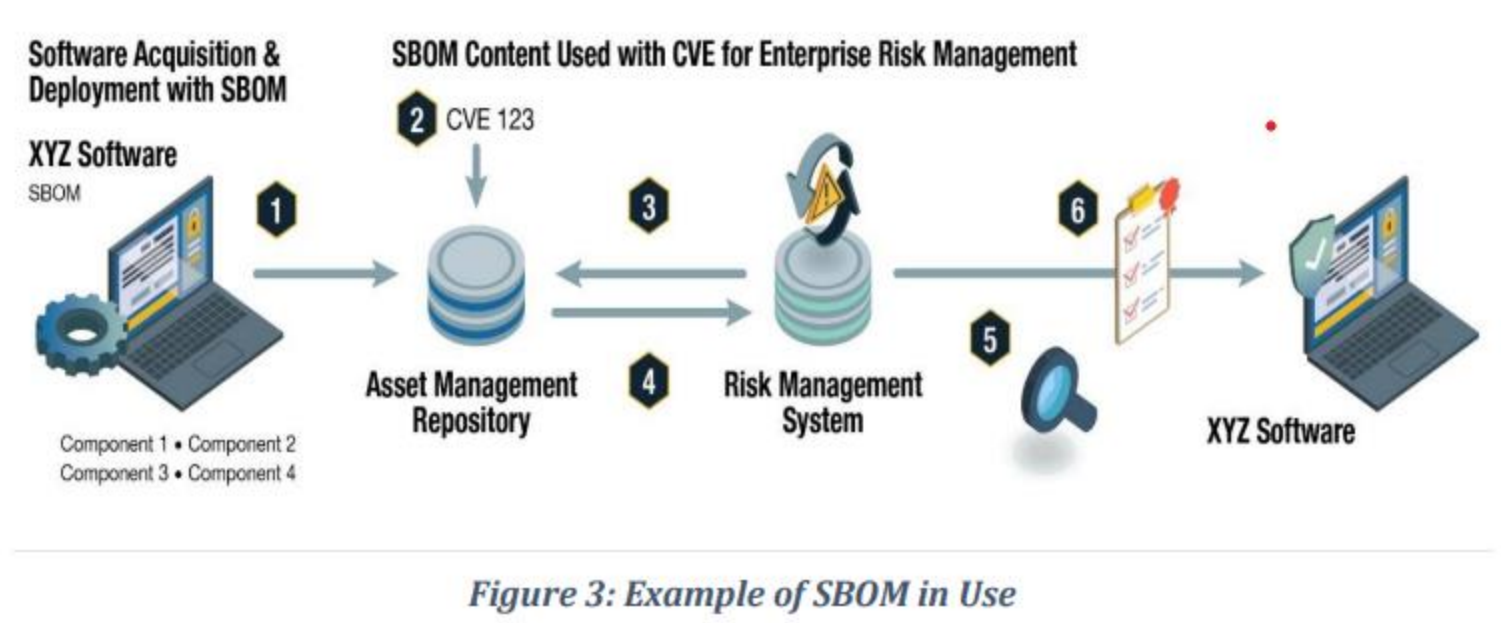


Figure 3: Example of SBOM in Use

Risk Identification

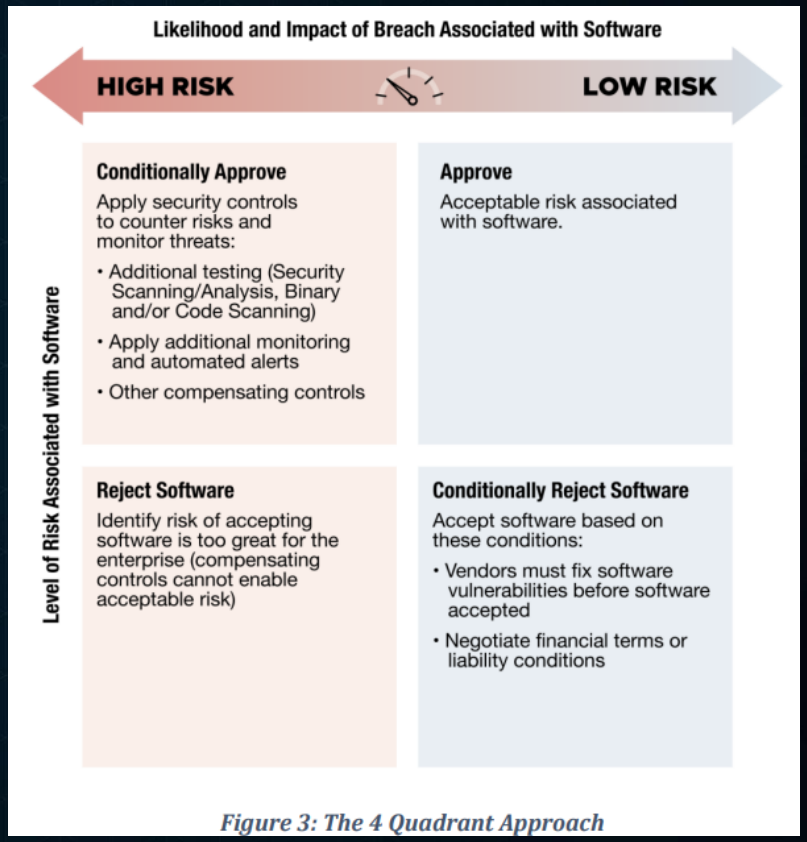


Figure 3: The 4 Quadrant Approach

Securing the Software Supply Chain: Recommended Practices Guide

1. SBOM Consumption
   - Component Information
   - Ingestion & Management for Enterprise
   - Mapping and Asset Management

3

2. Managing OSS & SBOMs
   - Evaluate OSS acceptance/deployment process
   - Protection of existing environment
   - Management of Identity and Access control
   - Update and distribution of SBOMs
   - Verification of package contents

Managing OSS & SBOMs

Ensuring Supply Chain integrity & security of software from production through delivery

- Examples Threats:
  - Exploitation of software design flaws
  - Infiltration of the supplier's network with malicious code prior to the final software product being delivered
  - Inclusion of vulnerable 3rd party components
- Example Mitigations:
  - Evaluate software using precursory analysis such as the National Vulnerability Database (NVD) to determine whether the software should be considered for selection
  - Follow established guidelines for maintenance and vulnerability responses for all products released
  - Monitor third-party sources for defects/links to ongoing vulnerability assessments
  - Before shipping software packages to customers, the developer or supplier should perform binary composition analysis to verify the contents of the package.

ENDURING SECURITY FRAMEWORK

SECURING THE SOFTWARE SUPPLY CHAIN RECOMMENDED PRACTICES FOR

Managing Open Source Software and Software Bill of Materials

Software incorporated and/or utilized through open source may have embedded issues. Therefore, particularly close attention should be paid as to how these modules are used and bundled with the software at release. Common methods of compromise used against software supply chains continue to include exploitation of software design flaws, incorporation of vulnerable third-party components into a software product, infiltration of the supplier's network with malicious code prior to the final software product being delivered, and injection of malicious software that is built and then deployed by the customer. Things to consider when evaluating the overall Open Source Software (OSS) acceptance process and deploying an open source solution into an existing product development environment include: its composition; process and procedures used when adopting open source software; and the management, tracking and distribution of approved software components using an SBOM.

THREATS

- Selection/Risk assessment
- Licensing/Export control
- Maintenance/Vulnerability responses and Secure Software Delivery

RECOMMENDED PRACTICES

- Evaluation of software using precursory analysis such as the National Vulnerability Database (NVD) to determine whether the software should be considered for selection.
- Follow established guidelines for maintenance and vulnerability responses for all products released.
- When adopting OSS, developers should extract export-related information such as cryptographic algorithms used in the OSS and any other cryptographic dependencies the OSS requires.
- All third-party source is monitored for defects and linked to ongoing vulnerability assessments in both the supplier and developer roles.
- Ongoing alerting and monitoring can occur in house by running the vulnerability scanning activity used within the ingestion process, or can be provided by external sources.
- Suppliers must perform code signing for all software and firmware in components that are delivered to external entities including customers and partners.
- Before shipping software packages to customers, the developer or supplier should perform binary composition analysis to verify the contents of the package.
- Finally, strategies outlined in the SSDF should be used to manage identity and access control with respect to the generation, signing, updating and distribution of SBOMS for specific products.

NSAESF@CYBER.NSA.GOV
CYBERCENTER.NSA.GOV

Securing the Software Supply Chain: Recommended Practices Guide

ESF SBOM Guidance mapped to the NIST SSDF activities

Table 1: Associated SSDF Activities

Section	SSDF Activity(ies) Implemented
2. Open-Source Software Management	• Prepare the Organization (PO)
3. Creating and Maintaining a Company Internal Secure Open-source Repository	• Protect the Software (PS) • Produce Well-Secured Software (PW) • Respond to Vulnerabilities (RV)
4. Maintenance, Support and Crisis Management	• Protect the Software (PS) • Respond to Vulnerabilities (RV)
5. SBOM Creation, Validation and Artifacts	• Protect the Software (PS) • Produce Well-Secured Software (PW) • Respond to Vulnerabilities (RV)

Securing the Software Supply Chain: Recommended Practices Guide

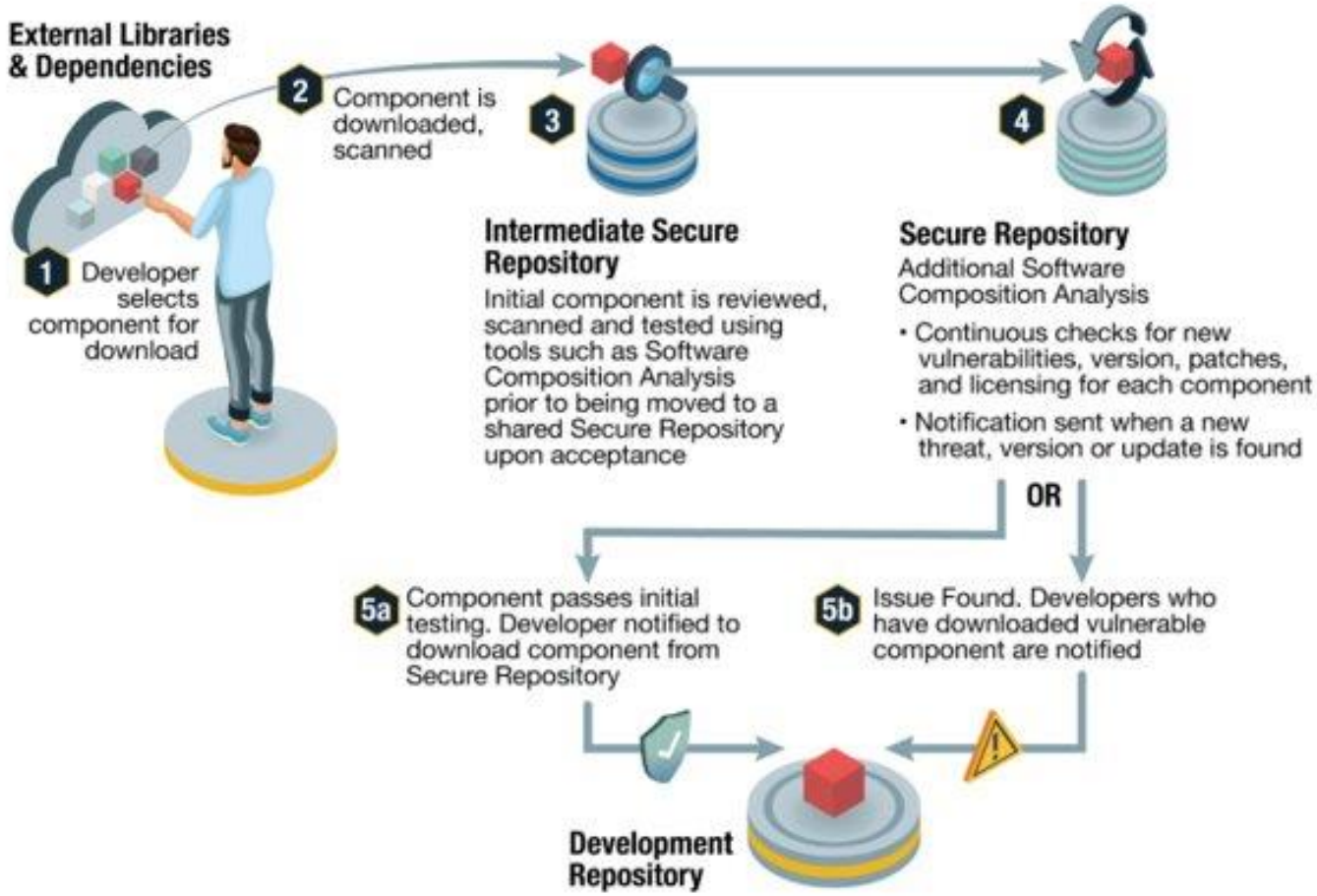


Figure 1: Maintaining Secure Open-Source Repository

Securing the Software Supply Chain: Recommended Practices Guide

Secure Supply Chain Consumption Framework (S2C2F)

- Ingest It
- Scan It
- Inventory It
- Update It
- Audit It
- Enforce It
- Rebuild It
- Fix It + Upstream It

Practice	Requirement ID	Maturity Level	Requirement Title	Benefit
Audit It	AUD-1	L3	Verify the provenance of your OSS	Able to track that a given OSS package traces back to a repo
	AUD-2	L2	Audit that developers are consuming OSS through the approved ingestion method	Detect when developers consume OSS that isn't detected by your inventory or scan tools
	AUD-3	L2	Validate integrity of the OSS that you consume into your build	Validate digital signature or hash match for each component
	AUD-4	L4	Validate SBOMs of OSS that you consume into your build	Validate SBOM for provenance data, dependencies, and its digital signature for SBOM integrity
Enforce It	ENF-1	L2	Securely configure your package source files (i.e., nuget.config, npmrc, pip.conf, pom.xml, etc.)	By using NuGet package source mapping, or a single upstream feed, or using version pinning and lock files, you can protect yourself from race conditions and Dependency Confusion attacks

Current SBOM Information

- Office of Management and Budget (OMB) Memorandum M-22-18 *Enhancing the Security of the Software Supply Chain through Secure Software Development Practices*[5-6] (Sept 2022)
  - Federal agencies must only use software from producers who can attest to complying with the NIST Guidance[7] (except agency developed, freely obtained and publicly available, see M-23-16)
    - Software includes firmware, operating systems, applications, and application services (e.g., cloud-based software), as well as products containing software
    - Required
      - obtain a self-attestation[8] from the software producer before using
      - or identify practices to which they cannot attest, document practices they have in place to mitigate those risks, and require a Plan of Action & Milestones (POA&M) to be developed
      - or provide a third-party assessment
    - Agencies may obtain conformance to secure software development practices such as a Software Bill of Materials (SBOMs)
- SPDX 3.0

From the Community-led Working Group on SBOM Tooling and Implementation, facilitated by Cybersecurity and Infrastructure Security Agency [cisa.gov/sbom]

Table 1: SBOM Type Definition and Composition

SBOM Type	Definition	Data Description
Design	SBOM of intended design of included components (some of which may not exist) for a new software artifact.	Typically derived from a design specification, RFP, or initial concept.
Source	SBOM created directly from the development environment, source files, and included dependencies used to build an product artifact.	Typically generated from software composition analysis (SCA) tooling, with manual clarifications.
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.	Typically generated as part of a build process. May consist of integrated intermediate Build and Source SBOMs for a final release artifact SBOM.
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.	Typically generated through analysis of artifacts by 3rd party tooling.
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.	Typically generated by recording the SBOMs and configuration information of artifacts that have been installed on systems.
Runtime	SBOM generated through instrumenting the system running the software, to capture only what is loaded and executing in memory, 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.	Typically generated from tooling interacting with a system to record the artifacts present in a running environment and/or that have been executed.

MITRE

© 2024 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 23-01614-54

Changes in SPDX 3.0

Bob M
Senior Software and Supply Chain Principal Engineer
Cyber Solutions Innovation Center
MITRE Labs

23 January 2024

MITRE | SOLVING PROBLEMS FOR A SAFER WORLD

© 2024 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 23-01614-54

3T-SBOM Usage Scenarios Around SBOMs

OMG CISQ

- Refer, Transfer or Purchase (definition of what it is)
- Pedigree (history of how it was produced)
- Provenance (chain of custody of it)
- Integrity (cryptographic basis of unalteredness)
- Proper and Legal (conditions about its use)
- Known Sw Vulns (known fixes are applied to it)
- Assurance (safe-secure-resilient)
- SBoM of a SW Service (SBoM of sw delivering service)
- Supply Chain Sequence Integrity

https://www.mitre.org/publications/technical-papers/standardizing-sbom-within-the-sw-development-tooling-ecosystem

© 2024 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 23-01614-54

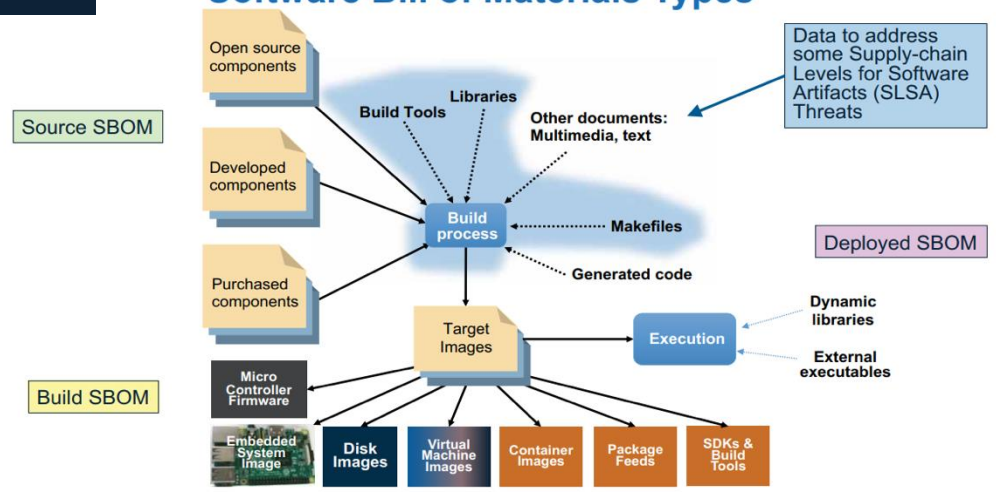
SPDX Profiles Overview

SPDX SECURITY	Security information - vulnerability details related to software
SPDX BUILD	Build related information - provenance and reproducible builds
SPDX AI	Information about AI models - ethical, security, and model data
SPDX DATA	Information about datasets - AI and other data use cases
SPDX LICENSING	Information about copyrights and licenses - supports compliance
SPDX SOFTWARE	Information specific to software
SPDX CORE	Information used across all profiles

MITRE

© 2024 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 23-01614-54

Software Bill of Materials Types



MITRE

© 2024 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 23-01614-54

Upcoming

- FAR-2021-0017[10], Published 10/03/24, Comments Close 12/4/2023 - Federal Acquisition Regulation: Cyber Threat and Incident Reporting and Information Sharing
  - Require contractors to develop and maintain a software bill of materials (SBOM) for any software used in the performance of the contract regardless of whether there is any security incident.
  - Provide access to the SBOM (and its updates) to the Government.
  - Machine-readable, industry-standard format
  - Comply with Section IV of National Telecommunications and Information Administration (NTIA) *The Minimum Elements For a Software Bill of Materials* (SBOM, July 12, 2021) published by the Department of Commerce[11]

- Sneak Peek: Information and Communications Technology (ICT) Supply Chain Risk Management Task Force[12] (ICT SCRM TF) Software Assurance Working Group – Draft *SwA Buyer's Guide* and worksheet. Attestation questions for
  - Supplier Governance
  - Software Supply Chain
  - Secure Software Development
  - Secure Software Deployment Controls
  - Vulnerability Management

- SPDX 3.0

SBOM Ongoing Work

- CISA is leading the advancement of SBOM work through many working groups https://cisa.gov/sbom; Hardware HBOM[13]; SBOM-a-RAMA[14-15]
- Devil is in the Details
  - Roll-up
  - Sharing
  - Automation tool results differ from developer to customer
  - Suppliers and proprietary data
  - Updates into closed networks
  - Identifying a vulnerability on an unclassified system often becomes classified
  - Who is responsible within each program, service and government organization to collect, validate and store
- Training for acquisition roles to support SBOMs
- Vulnerability Exploitability eXchange (VEX)

SBOM – Rock On!

- Get Started
  1. Create an SBOM:
     - Build tool support (Ex. Maven's works with tools to create a BOM centralizes a list of dependencies[16-17])
     - Main SBOM formats are SPDX & OWASP CycloneDx
  2. Use Software Composition Analysis (SCA) tools use SBOMs to check for license and vulnerability review
  3. Work to automate the above within your vulnerability management process
     - Continuously check items in the SBOM
     - Build a VEX where needed
     - Be able to provide the SBOM & VEX
     - Update to address issues found
     - Get updates to the customers

Questions?

NSAESF@CYBER.NSA.GOV
CAROL LEE – CALEE3@NSA.GOV