Workgroup: Network Working Group
Internet-Draft:
draft-birkholz-scitt-software-use-cases-00
Published: 25 October 2022
Intended Status: Informational
Expires: 28 April 2023
Authors: H. Birkholz (Fraunhofer SIT)
D. Brooks (REA)
R. Martin (MITRE)
B. Knight (Microsoft)

Detailed Software Supply Chain Uses Cases for SCITT

Abstract

Generalized Software Supply Chain Use Case Descriptions

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-birkholz-scitt-software-use-cases/.

Discussion of this document takes place on the SCITT Working Group mailing list (mailto:scitt@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/scitt/. Subscribe at https://www.ietf.org/mailman/listinfo/scitt/.

Source for this draft and an issue tracker can be found at https://github.com/ietf-scitt/draft-birkholz-scitt-software-supply-chain-use-cases.

Status of This Memo

Table of Contents

1. Introduction

Modern software applications are an intricate mix of first-party and third-party code, development practices and tools, deployment methods and infrastructure, and interfaces and protocols. The software supply chain comprises all elements associated with an application's design, development, deployment, and maintenance throughout its entire lifecycle. The complexity of software coupled with a lack of lifecycle visibility increases the risks associated with system attack surface and the number of cyber threats capable of harmful impacts, such as exfiltration of data, disruption of operations, and loss of reputation, intellectual property, and financial assets. There is a need for a platform architecture that will allow consumers to know that suppliers maintained appropriate

security practices without requiring access to proprietary intellectual property. SCITT-enabled products and analytics solutions will assist in managing compliance and assessing risk to help prevent and detect supply chain attacks across the entire software lifecycle while prioritizing data privacy.

1.1. Terminology

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Generic Problem Statement

Supply chain security is paramount to protecting critical infrastructure, aerospace, and defense and avoiding impacts on security, the economy, public health, and safety. It has historically focused on risk management practices to safeguard logistics, meet compliance regulations, demand forecasts, and optimize inventory. While these elements are foundational to a healthy supply chain, an integrated cyber security-based perspective of the software supply chains remains broadly undefined. Recently, the global community has experienced numerous supply chain attacks by cybercriminals targeting weaknesses in software supply chains. As illustrated in Figure 1, a software supply chain attack may leverage one or more lifecycle stages and directly or indirectly target the component.

generic supply chain threats diagram here

Figure 1: Example Lifecycle Threats

DevSecOps relies on third-party and open-source solutions, expanding supply chain complexity, and reducing the visibility of the lifecycle compliance. One solution approach is to enhance the auditability and accountability of Digital Supply Chain Artifacts (DSCA) by using an interoperable, scalable, and flexible decentralized architecture with a transparent registry. The required software artifacts are highly variable based on community policy requirements, and the solution approach should be artifact agnostic to enable adaptation to these broad policies. Example artifacts may include commit signatures, build environment and parameters, software bill of materials, static and dynamic application security testing results, fuzz testing results, release approvals, deployment records, vulnerability scan results, and patch logs.

3. Notational Implementation

TBD

deployment chain diagram here

Figure 2: Deployment Example of SCITT in Software Development

4. Software Supply Chain Use Cases

4.1. Software Bill of Material

Micro Coding Wizards (MCW) is a small, fictional, software development company providing software solutions to help manage a fleet of electric vehicles (EV) for corporations. MCW's software solution, MCWManager, is an asset management platform specifically designed to manage electric vehicle fleets. MCWManager tracks usage, charge level, range, and other important characteristics of each EV in the fleet. The US Department of the Interior (DOI), a government agency has expressed interest in licensing the MCWManager software to manage a fleet of 20 Electric Vehicles, spread across the Western Region, which includes States west of the Rocky Mountains.

MCW has been informed by DOI that their software will be subject to Cybersecurity Executive Order (EO) 14028 recommendations from NIST and will need to supply "Software Bill of Materials" (SBOM) and a "Vulnerability Disclosure Report" (VDR) NIST attestation to the DOI prior to procurement.

4.1.1. Producer (MCW) Actions

A software producer, in this case MCW, creates a digitally signed SBOM, listing all the components contained in the final "distribution package" which a software consumer downloads in preparation for deployment within their digital ecosystem. The following steps are performed by the Software Producer:

1. Create the "final SBOM" listing all the components contained in the final distribution package of a software product, which the customer will install into their environment. The SBOM must follow NTIA minimum elements for SBOM's and other NTIA and NIST recommendations for SBOM, to meet Executive Order 14028 and OMB M-22-18 requirements. (Co)SWID, SPDX or CycloneDX SBOM formats are acceptable for this artifact.

2. Digitally sign the SBOM artifact.

3. Place the SBOM and digital signature artifacts within an access-controlled location, i.e. a customer portal, and provide

the end consumer with a link to these artifacts for downloading to the customers environment.

4.1.2. Consumer Actions

A software consumer, in this case DOI, obtains a digitally signed SBOM artifact from a software vendor, which initiates the following risk assessment process:

1. Produce a SHA-256 hash value for the SBOM artifact.

2. Verify the digital signature over the SBOM artifact and identify the signing key used to sign the SBOM, referred to as the SKID (Secret Key ID), assuming the signature is verified successfully.

3. Submit an inquiry to a trusted SCITT Registry requesting confirmation that a trust declaration is present in the registry for the combination of SHA-256 hash value and SKID associated with the SBOM.

4. If a trust declaration is on file with the SCITT trusted registry then continue with the risk assessment, otherwise inform the consumer that the SBOM hash and SKID combination are not registered, and the risk assessment ceases.

5. Continue with the risk assessment by performing a vulnerability search for each SBOM component, identifying any CVE's that are reported.

deployment scenario diagram here

Figure 3: Potential SCITT Implementation Scenario

4.2. Vulnerability Disclosure Report

MCW has been informed by DOI that their software will be subject to Cybersecurity Executive Order (EO) 14028 recommendations from NIST and will need to supply "Software Bill of Materials" (SBOM) and a "Vulnerability Disclosure Report" (VDR) NIST attestation to the DOI prior to procurement.

4.2.1. Producer (MCW) Actions

A software producer, in this case MCW, participates in a Vulnerability Disclosure Program, and generates a Vulnerability Disclosure Report listing all the known vulnerabilities and mitigations, which a software consumer downloads in preparation for

deployment within their digital ecosystem. The following steps are performed by the Software Producer:

1. Participate in a Vulnerability Disclosure program.

2. Generate a Vulnerability Disclosure Report (VDR) listing all the known vulnerabilities and mitigation plans to meet Executive Order 14028 and OMB M-22-18 requirements.

3. Digitally sign the VDR artifact.

4. Place the VDR and digital signature artifacts within an access-controlled location, i.e., a customer portal, and provide the end consumer with a link to these artifacts for downloading to the customers environment.

4.2.2. Consumer Actions

A software consumer, in this case DOI, obtains a digitally signed VDR artifact from a software vendor, which initiates the following risk assessment process:

1. Produce a SHA-256 hash value for the VDR artifact.

2. Verify the digital signature over the VDR artifact and identify the signing key used to sign the VDR, referred to as the SKID (Secret Key ID), assuming the signature is verified successfully.

3. Submit an inquiry to a trusted SCITT Registry requesting confirmation that a trust declaration is present in the registry for the combination of SHA-256 hash value and SKID associated with the SBOM.

4. Consumer checks SBOM against NIST NVD. If vulnerabilities have been reported within NVD and not in the producer provided VDR, then raise an issue with the producer for report accuracy.

5. If a trust declaration is on file with the SCITT trusted registry then continue with the risk assessment, otherwise inform the consumer that the VDR hash and SKID combination are not registered, and the risk assessment ceases.

6. Continue with the risk assessment by performing a vulnerability search for each SBOM component, identifying any CVE's that are reported.

4.3. NIST self-attestation: SSDF Framework

Consistent with the NIST Guidance and by the timelines identified below, agencies are required to obtain a self-attestation from the software producer before using the software.

4.3.1. Producer (MCW) Actions

An acceptable self-attestation must include the following minimum requirements:

1. The software producer's name.

2. A description of which product or products the statement refers to (preferably focused at the company or product line level and inclusive of all unclassified products sold to Federal agencies).

3. A statement attesting that the software producer follows secure development practices and tasks that are itemized in the standard self-attestation form.

4. Self-attestation is the minimum level required; however, agencies may make risk-based determinations that a third-party assessment is required due to the criticality of the service or product that is being acquired, as defined in M-21-30.

4.3.2. Consumer Actions

TBD

5. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

Appendix A. TODO List

*Promotion Scenario: '3rd party lab validates the detail instead of their own test'

*Endorsement Scenario: Audit downstream independent of issuer and provide an endorsement

*CI/CD SCITT interaction - Create a model before talking to Github
(Statements about SW could be listed. Policy management can be
done via SCITT through SW development lifecycle)

Authors' Addresses

Henk Birkholz
Fraunhofer Institute for Secure Information Technology
Rheinstrasse 75
64295 Darmstadt
Germany

Email: henk.birkholz@sit.fraunhofer.de

Dick Brooks
REA

Email: dick@reliableenergyanalytics.com

Robert Martin
MITRE

Email: ramartin@mitre.org

Brian Knight
Microsoft

Email: brianknight@microsoft.com