OpenChain Security Assurance Specification 1.1 (Draft)

OpenChain Project: Building Trust In The Supply Chain Since 2016

Introduction

The OpenChain Project is working towards a supply chain where open source is delivered with trusted and consistent compliance information. As part of this mission, we maintain OpenChain ISO/IEC 5230:2020, the International Standard for open source license compliance. A natural next step in support of the broader mission was to develop a guide to identify and present the minimum core set of requirements every Security Assurance program should satisfy with respect to the use of Open Source software.

For context, OpenChain ISO/IEC 5230:2020 is a process management specification that identifies inbound, internal and outbound inflection points where a process, policy or training should exist. The identification and tracking of software used and deployed is an inherent part of getting this right, and this allows our approach to also be useful for security or export control.

The OpenChain Project community noticed OpenChain ISO/IEC 5230:2020 being used in the security domain and decided to develop this security specification to satisfy market demand. This specification is intended to identify and describe the key requirements of a quality Security Assurance Program in the context of using Open Source Software. It focuses on a narrow subset of primary concern: checking Open Source Software against publicly known security vulnerabilities like CVEs, GitHub/GitLab vulnerability reports, and so on.

The OpenChain security assurance specification focuses on the “what” and “why” aspects of a quality Security Assurance Program rather than delving into to “how” and “when.” This is a conscious decision to ensure flexibility for organizations of any size and in any market to use this reference specification. This approach, along with the types of processes identified, is built on more than half a decade of practical global feedback around the creation and management of such programs. The result is that a company can frame a program that precisely fits their supply chain requirements, scoped to a single product or a complete legal entity, and take this solution to market quickly and effectively.

This specification is built from the Security Assurance Reference Guide 2.0 (Release Candidate 1) published on 2022-03-28. That reference document went through a final approval process via our normal voting practice to transform into this published security specification. The scope of this reference specification may expand over time based on community feedback.

This introduction to this specification describes its purpose. Section 2 defines key terms used throughout this document. Section 3 defines the requirements that a Program must satisfy to achieve a core level of Security Assurance. Each requirement consists of one or more verification materials (i.e., records) that must be produced to satisfy the requirement.

Verification materials are not required to be made public, though an organization may choose to provide them to others, potentially under a Non-Disclosure Agreement (NDA).

OpenChain Security Assurance Reference Specification

1: Scope

This document specifies the key requirements of a quality Open Source Software Security Assurance Program that establishes trust between organizations exchanging software solutions comprised of Open Source Software.

2: Terms, Definitions and Examples

For the purposes of this document, the following terms and definitions apply.

2.1 - Component Records

A Component Record should consist of a Supplier Name, Component Name, Version of the Component, Other Unique Identifiers, Dependency Relationship, Author of SBOM Data and Timestamp (as per NTIA minimum elements for SBoM)

2.2 - Customer Agreement

There is agreement that the processes used to find, track or fix security issues are regarded as sufficient and correct by relevant customers or user organizations if applicable.

2.3 - CVE

Common Vulnerabilities and Exposures (CVE) is a public database of disclosed computer software security issues and flaws. When someone refers to a CVE, they mean a security flaw that's been assigned a CVE ID number within the database. The CVE database is sponsored by the US Department of Homeland Security (DHS) and the Cybersecurity and Infrastructure Security Agency (CISA).

2.4 - Documented Evidence

This is explicit stored information that outlines, explains or records information related to activities and actions referred to in this specification.

2.5 - Known Vulnerability

Security vulnerabilities previously discovered in Open Source Software components that are publicly available. That would include any publicly published vulnerabilities including but not limited to CVEs, GitHub/GitLab vulnerability alerts, package manager alerts and so forth.

2.6 - Newly Discovered Vulnerability

Security vulnerabilities just discovered in Open Source Software components that are publicly available. That would include any publicly published vulnerabilities including but not limited to CVEs, GitHub/GitLab vulnerability alerts, package manager alerts and so forth.

2.7 - Open Source Software

Software subject to one or more licenses that meet the Open Source Software Definition published by the Open Source Software Initiative (see www.opensource.org/osd) or the Free Software Definition published by the Free Software Foundation (see www.gnu.org/philosophy/free-sw.html).

2.8 - Program

The set of policies, processes and personnel that comprise an organization's security assurance activities.

2.9 - Program Participants

Any organization employee or contractor that defines, contributes to or has responsibility for preparing Supplied Software. Note: Depending on the organization, that may include (but is not limited to) software developers, release engineers, quality engineers, product marketing, product management and procurement.

2.10 - Security Assurance

The confidence that a system meets the requirements for security best practices and is resilient against Known Vulnerabilities.

2.11 - Security Testing

A process for the analysis of software (or other components) that allows for understanding their current and potential future management in the context of Known Vulnerabilities.

2.12 - Software Bill of Materials (SBOM)

Information in a structured format such as SPDX ISO/IEC 5962:2021 that allows the exchange of information for a software package, which could usefully include name, version, origin, license, copyright and Known Vulnerabilities in a manner useful to third parties.

2.13 - Supplied Software

Software that an organization distributes or makes available to third parties (e.g., other organizations or individuals).

2.14 - Verification Materials

Materials that demonstrate that a given requirement of the specification is satisfied.

3 - Requirements

3.1 - Program Foundation

3.1.1 - Policy

A written policy will be created that governs Open Source Software Security Assurance of Supplied Software. The policy will be internally communicated. The policy and its method of communication will have a review process to ensure they are current and relevant.

Verification Material(s):

- 3.1.1.1: A documented Open Source Software Security Assurance policy;
- 3.1.1.2: A documented procedure to make Program Participants aware of the Security Assurance policy.

Rationale:

This is to make sure a process exists to create, record, and make Program Participants aware of the existence of an Open Source Software Security Assurance policy. Although no requirements are provided here on what should be included in the policy, other sections may impose additional requirements.

3.1.2 - Competence

The organization shall:

- Identify the roles and responsibilities that impact the performance and effectiveness of the Program;
- Determine the necessary competence of Program Participants fulfilling each role;
- Ensure that Program Participants have appropriate education, training, and/or experience;
- Where applicable, ensure Program Participants take actions to acquire the necessary competence;
- Retain appropriate documented information as evidence of competence as well as who is currently a participant in the program.

Verification Material(s):

- 3.1.2.1: A documented list of roles with corresponding responsibilities for the different Program Participants;
- 3.1.2.2: A document that identifies the competencies for each role;
- 3.1.2.3: List of participants and their roles;
- 3.1.2.4: Documented evidence of assessed competence for each Program Participant;
- 3.1.2.5: Documented Evidence of periodic reviews and changes made to the process;
- 3.1.2.6: Documented verification that these processes are current with company internal best practices and who is assigned to accomplish them.

Rationale:

To ensure that Program Participants have a sufficient level of competence for their respective roles and responsibilities.

3.1.3 - Awareness

The organization will ensure that the Program Participants are aware of:

- The Open Source Software Security Assurance policy;
- Relevant Program objectives;
- Their contribution to the effectiveness of the Program;
- The implications of not following the Program's requirements.

Verification Material(s):

- 3.1.3.1: Documented Evidence of assessed awareness for the Program Participants - which should include the Program's objectives, one's contribution within the Program, and implications of Program non-conformance.

Rationale:

To ensure the Program Participants have obtained a sufficient level of awareness for their respective roles and responsibilities within the Program.

3.1.4 - Program Scope

A Program should have defined guiding principles and scope that match the risk management policy of the entire organization. It should be clear whether the Program applies to a product line, a department, or the entire organization. It should also be understood that this scope may change over time and metrics may be used to assess its ongoing effectiveness.

Verification Material(s):

- 3.1.4.1: A written statement that clearly defines the scope and limits of the Program;
- 3.1.4.2: A set of metrics the program shall achieve to improve;

- 3.1.4.3: Documented Evidence from each review, update, or audit to demonstrate continuous improvement.

Rationale:

To provide the flexibility to construct a Program that best fits the scope of an organization's needs. Some organizations could choose to maintain a Program for a specific product line while others could implement a Program to govern the Supplied Software of the entire organization.

3.1.5 - Standard Practice Implementation

The Organization demonstrates a sound and robust handling procedures of Known Vulnerabilities and Secure Software Development by defining and implementing following procedures:

- Method to identify structural and technical threats to the Supplied Software is defined;
- Method for detecting existence of Known Vulnerabilities in Supplied Software;
- Method for following up on identified Known Vulnerabilities;
- Method to communicate identified Known Vulnerabilities to customer base when warranted;
- Method for analyzing Supplied Software for newly published Known Vulnerabilities post release;
- Method for continuous and repeated Security Testing is applied for all Supplied Software before release;
- Method to verify that identified risks will have been addressed before release of Supplied Software;
- Method to export information about identified risks to third parties as appropriate.

A process shall exist for the Security Assurance methods listed above.

Verification Material(s):

- 3.1.5.1: A documented procedure exists for each of the methods identified above.

Rationale:

To ensure appropriate processes exists for detecting and following up on Known Vulnerabilities in the Supplied Software.

3.2 - Relevant Tasks Defined And Supported

3.2.1 - Access

Maintain a process to effectively respond to Known Vulnerability external inquiries. Publicly identify a means by which a third party can inquire about a Known Vulnerability with respect to a given software offering.

Verification material(s):

- 3.2.1.1: Publicly visible method to allow third parties to make Known Vulnerability or Newly Discovered Vulnerability enquires (e.g., via an email address or web portal that is monitored by Program Participants);
- 3.2.1.2: An internal documented procedure exists for responding to third party Known Vulnerability inquiries.

Rationale:

To ensure there is a reasonable way for third parties to contact securely the organization regarding security vulnerability inquiries and that the organization is prepared to respond.

3.2.2 - Effectively Resourced

Identify and Resource Program Task(s):

- Assign accountability to ensure the successful execution of Program tasks;
- Program tasks are sufficiently resourced;
- Sufficient time to perform the tasks have been allocated; and
- Adequate funding has been allocated;
- A process exists for reviewing and updating the policy and supporting tasks;
- Technical expertise pertaining to Known Vulnerabilities is accessible to those who may need such guidance.

Verification Material(s):

- 3.2.2.1: Document with name of persons, group or function in Program role(s) identified;
- 3.2.2.2: The identified Program roles have been properly staffed and adequate funding provided;
- 3.2.2.3: Identification of expertise available to address identified Known Vulnerabilities;
- 3.2.2.4: A documented procedure that assigns internal responsibilities for Security Assurance.

Rationale:

To ensure: i) Program responsibilities are effectively supported and resourced and ii) policies and supporting processes are regularly updated to accommodate changes in Security Assurance best practices.

3.3 - Open Source Software Content Review And Approval

3.3.1 - Software Bill of Material (SBOM)

A process shall exist for creating and maintaining a bill of materials that includes each Open Source Software component from which the Supplied Software is comprised.

Verification Material(s):

- 3.3.1.1: A documented procedure ensuring for all Open Source Software used in the Supplied Software is continuously recorded across the lifecycle of the Supplied Software. This may include an archive of all Open Source Software used in the Supplied Software;
- 3.3.1.2: Open Source Software Component Records for the Supplied Software that demonstrates the documented procedure was properly followed.

Rationale:

To ensure a process exists for creating and managing a Software Bill of Materials used to construct the Supplied Software. A bill of materials is needed to support the systematic review of each component to understand if any Known Vulnerabilities exist

3.3.2 - Security Assurance

- For each Open Source Software component in the bill of materials for the Supplied Software release under review;
- Apply method for detecting existence of Known Vulnerabilities;
- For each identified Known Vulnerability assign a risk/impact score;
- For each detection and assigned score determine and document necessary remediation steps suitable for the use-case of the software and get Customer Agreement at or above a previously determined level (i.e., all severity scores above 4.5 …);
- Depending on the risk/impact score take the appropriate action (e.g., contact customers if necessary, upgrade Software Component, no further action, …);
- If a Newly Discovered Vulnerability is present in previously distributed Supplied Software, depending on the risk/impact score take the appropriate action (e.g., contact customers if warranted);
- An ability to monitor Software Components after their release to market and to respond to Known Vulnerability or Newly Discovered Vulnerability disclosures.

Verification Material(s):

- 3.3.2.1: A documented procedure for handling detection and resolution of Known Vulnerabilities for the Open Source Software components of the Supplied Software;
- 3.3.2.2: For each Open Source Software component a record is maintained of the identified Known Vulnerabilities and action(s) taken (including even if no action was required).

Rationale:

To ensure the Program is sufficiently robust to handle the identified Known Vulnerabilities for the Open Source Software from which the Supplied Software is comprised. That a procedure exists to support this activity and that the procedure is followed.

3.4 - Adherence To The Guideline Requirements

3.4.1 - Completeness

For a Program to be deemed conformant with this specification, the organization shall affirm that the Program satisfies the requirements presented in this document.

Verification Material(s):

- 3.4.1.1: Documented Evidence affirming the Program specified in §3.1.4 satisfies all the requirements of this document.

Rationale:

To ensure that if an organization declares that it has a Program that is conforming, that the Program has met all the requirements of this document. The mere meeting of a subset of these requirements is not considered sufficient.

3.4.2 - Duration

A Program that is conformant with this version of the specification will have a review period as follows: 18 months from the first certification, 24 months from the second certification and 36 months from the third certification. It will require review every 36 months after this.

Verification Material(s):

- 3.4.2.1: A document affirming the Program meets all the requirements of this specification, within the past 18 months of obtaining conformance validation.

Rationale:

It is vital for a Program to remain current with the specification requirements if an organization wants to assert conformance over time. This requirement ensures that the Program's supporting processes and controls do not erode if an organization continues to assert Program conformance over time.