Federal Office
for Information Security

BSI TR-03185-2

Secure Software Lifecycle for Open Source Software

Version 1.1.0

Document history

Table 1: Changelog

Version	***Date***	***Editor***	***Description***
0.1.0	16 June 2025	jdl, fvs	Initial draft.
1.0.0	18 July 2025	jdl, fvs	Feedback incorporated.
1.1.0	18 August 2025	jdl, fvs	Feedback incorporated.

German Federal Office for Information Security (BSI)
P.O. Box 20 03 63
53133 Bonn
Germany
Tel.: +49 800-274-1000 (Service Center)
E-Mail: oss-lifecycle@bsi.bund.de
Internet: https://bsi.de/EN/

Table of Contents

List of Tables 4
1 Introduction 5
1.1 Objectives 5
2 Structure of this Technical Guideline 6
2.1 Requirements Language 6
2.2 Definition of Terms 6
2.3 Methodology 7
3 Requirements 8
3.1 Governance (GV) 8
3.2 Legal (LE) 9
3.3 Quality (QA) 9
3.4 Build and Release (BR) 9
3.5 Vulnerability Management (VM) 10
3.6 Decommissioning (DE) 10
Glossary 11

List of Tables

Table 1: Changelog 2
Table 2: Modal verbs 6
Table 3: Definition of terms 6
Table 4: Governance requirements 8
Table 5: Legal requirements 9
Table 6: Quality requirements 9
Table 7: Requirements for building and releasing software 9
Table 8: Requirements for vulnerability management 10
Table 9: Requirements for the decommissioning of software 10

1 Introduction

Secure software is essential for the use of IT products in governments, businesses and societies. The German Federal Office for Information Security (BSI) appeals to all relevant stakeholders to consider information security from the outset and to ease estimating a software component's security posture as far as possible in order to enable its secure use.

This Technical Guideline BSI TR-03185-2 "Secure Software Lifecycle for Open Source Software", or "Secure OSS Lifecycle" for short, complements the BSI TR-03185-1 "Secure Software Lifecycle", which solely addresses the lifecycle of proprietary software products. Due to the fundamental differences in roles, the development process proper, and because of the intrinsic properties of Open Source Software licences (as defined by FSF[1], DFSG[2] and OSI[3]; see also SPDX[4] and openCode[5]), a "Secure Software Lifecycle" modelled for proprietary software does not fit. Hence this Technical Guideline was developed which focusses specifically on "Open Source Software (OSS)", also known as "Free / Libre and Open Source Software (FLOSS / FOSS)".

The European Union (EU) legislated the Cyber Resilience Act (CRA) to ensure cybersecurity of IT products throughout their entire lifecycle by obligating manufacturers and Open Source Software stewards to conduct appropriate risk assessments and to maintain an adequate level of information security for their software products etc. (i.e., "products with digital elements").

1.1 Objectives

This Technical Guideline

- is based on well recognised guidelines and frameworks (see section 3),
- is agnostic of tooling, development processes and available resources, and
- aims to be minimal and avoids placing additional burden on developers and project maintainers.

This Technical Guideline is intended to not contradict (in the sense of "to be compatible with")

- EU Cyber Resilience Act (CRA), and
- BSI IT-Grundschutz (IT basic protection).

[1] http://www.gnu.org/philosophy/free-sw.en.html#fs-definition;
German translation: http://www.gnu.org/philosophy/free-sw.de.html#fs-definition
[2] https://wiki.debian.org/DebianFreeSoftwareGuidelines;
German translation (unauthorized): https://de.wikipedia.org/wiki/Debian_Free_Software_Guidelines
[3] https://opensource.org/osd;
German translation (unauthorized):
https://de.wikipedia.org/wiki/Open_Source_Initiative#Definition_von_Open_Source
[4] https://spdx.org/licenses/
[5] https://opencode.de/de/wissen/rechtssichere-nutzung/open-source-lizenzen

2 Structure of this Technical Guideline

This chapter outlines the methodology of this Technical Guideline and the terms used.

2.1 Requirements Language

In the requirements, the modal verbs “SHOULD” and “MUST” written in capital letters are used in their respective forms and the respective negations to make clear how the respective verbs should be interpreted. The definitions used here are based on RFC2119[6] and DIN 820-2[7]:2020, Annex H.

Table 2: Modal verbs

Verb	***Definition***
MUST	This term means that this is a requirement that must be imperatively fulfilled (absolute requirement).
MUST NOT	This term means that something must not be done in no case (absolute prohibition).
SHOULD	This term means that a requirement usually must be fulfilled, but that there can be reasons to not fulfil the requirement. However, this must be carefully assessed and well founded.
SHOULD NOT	This term means that something should not be done usually, but that there can be reasons to do it. However, this must be carefully assessed and well founded.
MAY	This expression denotes a requirement that is optional.

2.2 Definition of Terms

Table 3: Definition of terms

Term	***Definition***
Manufacturer	A natural or legal person who develops or manufactures products with digital elements or has products with digital elements designed, developed or manufactured, and markets them under its name or trademark, whether for payment, monetization or free of charge.
Steward	A legal person, other than a manufacturer, that has the purpose or objective of systematically providing support on a sustained basis for the development of specific products with digital elements, qualifying as Free and Open Source Software and intended for commercial activities, and that ensures the viability of those products.
Maintainer	A person responsible for managing and overseeing a software project, including but not limited to writing code, reviewing contributions and ensuring the project’s quality and direction.
Contributor	A person (acting as independent or on behalf of an organization) who participates in the development of a software project by providing code, documentation, bug reports, or other forms of support.
Upstream	Refers to a repository/organization where a software project originates from and where most of the development takes place.

[6] https://www.rfc-editor.org/info/rfc2119
[7] https://de.wikipedia.org/wiki/DIN_820

Term	*Definition*
Downstream	A repository/organization other than upstream, where a software is enhanced, built, distributed or integrated.

2.3 Methodology

As outlined before, one goal of this Technical Guideline is to improve the security of software generally referred to as “Open Source Software (OSS)” or “Free / Libre and Open Source Software (FLOSS / FOSS)” without putting unnecessary burden on maintainers and contributors. In many OSS projects, additional work to comply with sophisticated security guidelines (such as creating and maintaining specialised documentation) will take away time that could be spent in applying practical security measures such as looking for and fixing vulnerabilities. Security incidents such as the backdoor in *xz-utils* or flaws in *curl* or *OpenSSL* (“Heartbleed”) show that there is a delicate balance to be met in order to successfully raise software security in environments with very little available resources. Hence this Technical Guideline does not appoint a specific party or role responsible for making and keeping an OSS project compliant to this Technical Guideline. In accordance with the CRA, this provides several options for who may implement these requirements (in order of preference):

1. Manufacturers or stewards depending on a software component within the scope of this guideline, should become involved in the “upstream” development to raise the software security of these components at their source.

2. Upstream project maintainers may choose to implement these requirements, likely motivated to raise the software security of their project.

3. Manufacturers or stewards of products depending on a software component within the scope of this guideline may maintain a “downstream” fork and implement these requirements there.

3 Requirements

The requirements in this section have been compiled from or are based on the following guidelines, guidance and frameworks:

- BSI IT-Grundschutz (ITG), Kompendium 2023

  https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/IT-Grundschutz-Kompendium/it-grundschutz-kompendium_node.html

- BSI TR-03185: Secure Software Lifecycle (TR), Version 1.0

  https://www.bsi.bund.de/dok/TR-03185-en

- Cyber Resilience Act (CRA), 2024/2847

  https://eur-lex.europa.eu/eli/reg/2024/2847/oj

- Open Chain (OC, ISO/IEC 18974), Edition 1, 2023

  https://openchainproject.org/security-assurance

- Open CRE (OCRE), unversioned, visited May 2025

  https://opencre.org/root_cres

- OpenSSF Security Baseline (OSPS), Version 2025-02-25

  https://baseline.openssf.org/versions/2025-02-25

- Secure Software Development Framework (SSDF, NIST SP 800-218), February 2022

  https://csrc.nist.gov/pubs/sp/800/218/final

- Supply-chain Levels for Software Artifacts (SLSA), Version 1.1

  https://slsa.dev/spec/v1.1/

The term “Induced by” in the rightmost column of the following tables expresses the fact that the requirements in this Technical Guideline are neither directly derived from nor equivalent to the linked references. Deep-links are provided where technically feasible. Readers may refer to these links to assist in guiding and informing implementation or contextual understanding of the Requirement. Furthermore, all text indicated as “Note” is non-normative.

3.1 Governance (GV)

Table 4: Governance requirements

ID	*Requirement*	*Induced by*
GV.01	Information on how to contribute to the project MUST be documented. Information about the expected quality of contributions SHOULD be given.	• OC: 4.1.2 • OSPS-GV-03

ID	Requirement	Induced by
GV.02	The project's repository, websites and sensitive data MUST be protected against unauthorized actions.	• OSPS-AC-01 • CRA: Annex I, Part I, No. (2)(d-f) • SSDF: PS.1.1

3.2 Legal (LE)

Table 5: Legal requirements

ID	Requirement	Induced by
LE.01	A license MUST be stated for all content made available by the project, including the project's documentation.	• OSPS-LE-02
LE.02	A copy of all licenses in use, or references hereto, MUST be provided.	• OSPS-LE-03

3.3 Quality (QA)

Table 6: Quality requirements

ID	Requirement	Induced by
QA.01	A list of third-party components used in the software MUST be available.	• OSPS-QA-02, OSPS-DO-06 • OCRE: 863-521
QA.02	All project's source code MUST be publicly readable.	• OSPS-QA-01
QA.03	The project MUST inform how to report defects.	• OSPS-DO-02, OSPS-GV-02 • CRA: Annex II, No. 2
QA.04	Procedures for testing MUST be implemented and utilised. Note: This can include, among others, a CI/CD pipeline for automated testing or a test harness for testing locally.	• OSPS-GV-03, OSPS-QA-06
QA.05	The project SHOULD take measures to reduce or avoid memory safety issues. Note: This can include additional instructions for memory safety in the contributions guide.	• OSPS-GV-03, OSPS-QA-06 • SSDF: PW.5.1, PW.6.1, PW.6.2
QA.06	All changes to the source code SHOULD be peer-reviewed.	• OSPS-AC-03 • SSDF: PW.2.1

3.4 Build and Release (BR)

The following requirements do not imply that any software has to be built or released.

Table 7: Requirements for building and releasing software

ID	Requirement	Induced by
BR.01	Information on how to build all software assets MUST be publicly available.	• OSPS-DO-01, OSPS-DO-03 • SLSA 1.1 Build L1

ID	Requirement	Induced by
BR.02	All releases and released software assets that are intended for use MUST be assigned unique, monotonically increasing version identifiers. Note: The version identifier must be unique, incrementally increased, and convey the context of the release as compared to past releases, e.g., within a specific branch.	• OSPS-BR-02 • CRA: Annex I, Part I, No. (2)(f)
BR.03	All assets MUST be distributed in a way that maintains integrity or, at least, allows the verification thereof. Means to maintain and verify the authenticity of the distributed assets MAY be deployed.	• OSPS-DO-03, OSPS-BR-03, OSPS-BR-06 • CRA: Annex I, Part I, No. (2)(d-f, i-k)
BR.04	All releases MUST provide a descriptive log of functional and security modifications.	• OSPS-BR-04 • CRA: Annex I, Part I, No. (2)(l); Annex I, Part II, No. 4; Annex II, No. 8(b)
BR.05	Released source packages MUST NOT contain any content that is not present in the project's repository or cannot be deterministically generated from that repository.	• OSPS-QA-05
BR.06	Builds SHOULD be reproducible.	• SLSA 1.1 Build L1

3.5 Vulnerability Management (VM)

Table 8: Requirements for vulnerability management

ID	Requirement	Induced by
VM.01	The project documentation MUST contain security contacts for reporting vulnerabilities. There SHOULD be a way to do that privately.	• OSPS-VM-01, OSPS-VM-02, OSPS-VM-03 • CRA: Annex I, Part I, No. (2)(c); Annex I, Part II, No. (5-7); Annex II, No. (1-3)
VM.02	The project MUST publish information about discovered vulnerabilities within a reasonable period of time.	• CRA: Annex I, Part II, No. (1, 4, 6) • OSPS-VM-04

3.6 Decommissioning (DE)

Table 9: Requirements for the decommissioning of software

ID	Requirement	Induced by
DE.01	The discontinuation of the project itself, parts of it, or specific versions or version ranges SHOULD be communicated adequately.	• TR: PROD.DECOM.2 • OSPS-DO-04, OSPS-DO-05
DE.02	Migration paths MAY be outlined.	• TR: PROD.DECOM.2

Glossary

All definitions are either taken directly from or compatible with (less specific) those given by the OpenSSF Security Baseline under https://baseline.openssf.org/versions/2025-02-25#lexicon.

Change

> “Any alteration of the project's codebase, … or documentation. This may include addition, deletion, or modification of content.”

Defect

> “Errors or flaws in the software that cause it to produce incorrect or unintended results, or to behave in an unintended way. Defects can include bugs, vulnerabilities, or other issues that impact the software's functionality or security. Defects may have originally been intentional, but a change in environment or understanding has made them undesirable.”

License

> “A legal document that defines the terms under which the software can be used, modified, and distributed.”

Project Documentation

> “Written materials related to the project, such as user guides, developer guides, and contribution guidelines. These documents help users and developers understand, contribute to, and interact with the software. At release time, this may include provenance information, licensing details, and other metadata.”

Release

> “*(verb)* The process of making a bundle of assets available to users [with the intention of it being used] … .”
>
> “*(noun)* A bundle … of code, documentation, and other assets made available to users [with the intention of it being used].”

Released Software Asset

> “Deliverables provided to users as part of a release.”

Repository

> “A storage location, [often] managed by a version control system, where the project's code, documentation, and other resources are stored. It [may] track changes, manage collaborator permissions, and include configuration options such [as for] … access controls.”

Version Identifier

> “A label assigned to a specific release of the software … . Commonly recommended formats are Semantic Versioning or Calendar Versioning.”

Vulnerability Reporting

> “The act of identifying and documenting exploitable vulnerabilities in released software assets. This may include privately or openly reporting vulnerabilities to maintainers, security teams, or the public, as well as tracking the resolution of these vulnerabilities.”