

経済産業省 御中

令和 3 年度 サイバー・フィジカル・セキュリティ対策促進事業
SBOM を導入・活用するサプライチェーン
モデルの構築に向けた調査・実証事業
報告書

2022 年 3 月

MRI 三菱総合研究所

目次

1. 背景と目的	1
2. 産業活動においてソフトウェアを安全に利活用するための枠組みに関する取組動向等の調査	2
2.1. NTIA 等の米国政府機関における取組動向の調査	2
2.1.1. 米国大統領令に関連する取組動向の調査	2
2.1.1.1. Executive Order on Improving the Nation’s Cybersecurity (EO 14028)	4
2.1.1.2. Definition of Critical Software Under Executive Order (EO) 14028	5
2.1.1.3. Security Measures for “EO-Critical Software” Use Under Executive Order (EO) 14028	8
2.1.1.4. Recommended Minimum Standards for Vendor or Developer Verification (Testing) of Software Under Executive Order (EO) 14028	11
2.1.1.5. H.R.4611 - DHS Software Supply Chain Risk Management Act of 2021	13
2.1.1.6. The Minimum Elements For a Software Bill of Materials (SBOM)	13
2.1.1.6.1. 追加で推奨されるデータフィールド	16
2.1.1.6.2. クラウドベースソフトウェアや SaaS の課題	17
2.1.1.6.3. SBOM の整合性や完全性の担保	17
2.1.1.6.4. 脆弱性と SBOM の関係	17
2.1.1.6.5. 依存関係における脆弱性とエクスプロイト可能性	17
2.1.1.6.6. レガシーソフトウェアのバイナリ解析	18
2.1.1.6.7. SBOM の柔軟性と統一性	18
2.1.1.7. Protecting Critical Software Through Enhanced Security Measures	18
2.1.1.8. SP 800-161 (Cybersecurity Supply Chain Risk Management (C-SCRM) Practices for Systems and Organizations) Rev. 1 (Draft)	19
2.1.1.9. Readout of White House Meeting on Software Security	20
2.1.1.10. SP 800-218 Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities	21
2.1.1.11. Recommended Criteria for Cybersecurity Labeling of Consumer Software	22
2.1.1.12. Software Supply Chain Security Guidance Under Executive Order (EO) 14028 Section 4e	23
2.1.1.13. OMB Statement on “Enhancing The Security Of Federally Procured Software”	24
2.1.2. NTIA の Software Component Transparency に関連する取組動向の調査	25
2.1.2.1. SBOM Tool Classification Taxonomy	26
2.1.2.2. SBOM at a Glance	28
2.1.2.3. Energy Sector SBOM PoC Charter	28
2.1.2.4. Vulnerability-Exploitability eXchange (VEX) – An Overview	29
2.1.2.5. How-To Guide for SBOM Generation	30
2.1.2.5.1. SBOM 生成のための情報収集方法	30
2.1.2.5.2. 具体的な SBOM 生成方法	31
2.1.2.6. Healthcare SBOM Proof of Concept – Phase II Summary	33
2.1.2.7. SBOM Myths vs. Facts	34
2.1.2.8. Software Suppliers Playbook: SBOM Production and Provision	36
2.1.2.8.1. SBOM 作成手順	36
2.1.2.8.2. SBOM 作成にあたって考慮すべき事項	37

2.1.2.8.3. SBOM に関する補足事項	37
2.1.2.9. Software Consumers Playbook: SBOM Acquisition, Management, and Use	38
2.1.2.9.1. サプライヤーから SBOM を取得する際の注意点	38
2.1.2.9.2. SBOM 活用のプロセス及びプラットフォーム	38
2.1.2.9.3. SBOM の知的財産及び機密保持	39
2.1.3. その他の米国政府機関における取組動向の調査	39
2.1.3.1. CISA における取組	39
2.1.3.1.1. Defending Against Software Supply Chain Attacks	40
2.1.3.1.2. Bad Practices	41
2.1.3.1.3. Binding Operational Directive 22-01: Reducing the Significant Risk of Known Exploited Vulnerabilities	41
2.1.3.1.4. SBOM-a-rama	41
2.1.3.2. FDA における取組	43
2.1.3.3. DoD における取組	46
2.2. 企業や業界団体、OSS コミュニティにおけるソフトウェア管理の取組動向の調査	46
2.2.1. 公開情報調査結果	47
2.2.1.1. Software ISAC における取組	47
2.2.1.2. Linux Foundation における取組	47
2.2.1.2.1. Open Source Management & Strategy Program	47
2.2.1.2.2. sigstore Project	48
2.2.1.2.3. Security Scorecards 2.0	49
2.2.1.2.4. Secure Open Source (SOS) Rewards Pilot Program	50
2.2.1.2.5. Alpha-Omega Project	50
2.2.1.3. Google における取組	51
2.2.1.3.1. Open Source Vulnerabilities (OSV) Project	51
2.2.1.3.2. Open Source Insight Project	51
2.2.1.4. 欧州 ENISA における取組	51
2.2.1.5. 英国における取組	53
2.2.1.6. フランス Etalab における取組	53
2.2.2. ヒアリング調査結果	54
2.2.2.1. トヨタ自動車株式会社における取組	55
2.2.2.2. 日本電気株式会社（NEC）における取組	55
2.2.2.3. 富士通株式会社における取組	55
2.3. OSS を含むソフトウェアの安全な利活用に関する基礎調査	55
2.4. 米国 CISA との勉強会の開催	59
3. OSS の利活用等に関する事例の調査	61
3.1. 事例集の拡充に関する観点整理	61
3.2. 公開文献調査	62
3.2.1. 公開文献事例 1（オリンパス株式会社）	63
3.2.2. 公開文献事例 2（Zendesk, Inc.）	63
3.2.3. 公開文献事例 3（Verizon Communications Inc.）	64
3.2.4. 公開文献事例 4（JS Foundation）	65

3.2.5. 公開文献事例 5 (Applause App Quality, Inc.)	65
3.2.6. 公開文献事例 6 (APRESIA Systems 株式会社)	66
3.2.7. 公開文献事例 7 (ウイングアーク 1st 株式会社)	66
3.2.8. 公開文献事例 8 (株式会社システムインテグレータ)	67
3.3. ヒアリング調査	69
3.4. 事例集の拡充の結果整理	71
4. ソフトウェアの利活用に係るセキュリティリスク、課題及び対応策の検討	72
4.1. 動向等に関する文献調査	72
4.1.1. OSS 活用における課題、失敗事例等	72
4.1.1.1. OpenSSL Heartbleed 事例	73
4.1.1.1.1. OpenSSL の概略	73
4.1.1.1.2. OpenSSL Heartbleed の概要	73
4.1.1.1.3. OpenSSL Heartbleed 脆弱性の影響	74
4.1.1.1.4. 自由度の高さと脆弱性	74
4.1.1.2. ShellShock (bash の脆弱性) 事例	75
4.1.1.2.1. UNIX 系システムにおけるシェルの位置づけ	75
4.1.1.2.2. bash の脆弱性 (ShellShock) 概要	75
4.1.1.2.3. 潜在的な脆弱性が長期間存在	76
4.1.1.3. Apache Struts2 脆弱性事例	77
4.1.1.3.1. Apache Struts 2 の概略	77
4.1.1.3.2. Apache Struts 2 の脆弱性 (CVE-2017-5638) 概要	77
4.1.1.3.3. CVE-2017-5638 の影響	77
4.1.1.3.4. 緊急対応と時間的制約の問題点	78
4.1.2. OSS 活用における課題解決、成功事例等	79
4.1.2.1. Linux ディストリビューションにおける OpenSCAP の活用	79
4.1.2.1.1. SCAP	79
4.1.2.1.2. OpenSCAP	79
4.1.2.2. OpenSCAP を応用して一般化した OSS 活用に結び付ける方法論	82
4.1.2.2.1. GNU/Linux における SCAP 利用	82
4.1.2.2.2. 特定のソフトウェア向け SCAP Security Guide	83
4.1.3. サプライチェーンにおける OSS 管理	83
4.1.3.1. Linux ディストリビューションのメンテナンス方法	83
4.1.3.1.1. GNU/Linux のディストリビューション	84
4.1.3.1.2. メンテナンスからみたディストリビューション	84
4.1.3.1.3. パッケージ管理システムによる管理の効率化	86
4.1.3.2. OpenChain プロジェクト (ISO/IEC 5230:2020)	87
4.1.3.2.1. OpenChain のセキュリティへの拡張	87
4.1.3.2.2. OpenChain プロジェクトと連動するプロジェクト	88
4.1.4. 脆弱性情報等の共有及び共有体制	88
4.1.4.1. CVE 及び NVD	88
4.1.4.1.1. CVE プログラム	88
4.1.4.1.2. NVD	89
4.1.4.1.3. OSS における CVE/NVD の傾向	90

4.1.4.2.	日本国内の共有体制及び JVN.....	91
4.1.4.2.1.	国内における脆弱性情報共有体制.....	91
4.1.4.2.2.	製品の脆弱性のコーディネーション.....	91
4.1.4.2.3.	脆弱性情報の公開のタイミング.....	92
4.1.4.2.4.	OSS における脆弱性対応の課題.....	93
4.1.4.3.	CVE と CNA の関係及び Sub-CNA の役割.....	95
4.1.4.3.1.	CNA と Sub-CNA.....	95
4.1.4.3.2.	Root CNA としての JPCERT/CC と国内 Sub-CNA.....	96
4.1.4.3.3.	OSS と CNA の関係について.....	97
4.1.5.	OSS コミュニティにおける連携・脆弱性流通の現状.....	98
4.1.5.1.	OSS 脆弱性管理支援プロジェクト OSV.....	98
4.1.5.1.1.	OSV とは.....	98
4.1.5.1.2.	「知る、防ぐ、修正する」のフレームワーク.....	98
4.1.5.1.3.	OSV の脆弱性情報提供サイト osv.dev.....	100
4.1.5.2.	バグバウンティプログラムと脆弱性情報流通.....	101
4.1.5.2.1.	バグバウンティプログラム例.....	101
4.1.5.2.2.	バグバウンティプログラムと脆弱性情報流通.....	102
4.2.	OSS を活用におけるセキュリティリスクへの有効性の高い対応策.....	104
5.	SBOM を導入や活用するサプライチェーンモデルの構築に向けた調査・実証.....	105
5.1.	目的.....	105
5.2.	SBOM 導入・活用に関する課題・障害等の整理.....	105
5.3.	SBOM 実証の設計・計画.....	106
5.3.1.	SBOM 実証に関する要件.....	106
5.3.2.	SBOM の有効性の評価の考え方.....	106
5.3.3.	SBOM 実証の対象と体制の検討.....	108
5.3.4.	SBOM 実証の構成と設計.....	109
5.3.4.1.	実証の基本構成.....	109
5.3.4.2.	実証項目の設計.....	110
5.3.5.	SBOM 実証の実施スケジュール.....	115
5.4.	SBOM 実証の実施結果.....	116
5.4.1.	実証の手順.....	116
5.4.1.1.	コストの測定.....	116
5.4.1.2.	効果の測定.....	120
5.4.2.	独自形式による部品管理.....	122
5.4.2.1.	コスト評価（独自形式）.....	122
5.4.2.1.1.	初期コスト（独自形式）.....	122
5.4.2.1.2.	運用コスト（独自形式）.....	123
5.4.3.	SBOM による部品管理.....	124
5.4.3.1.	コスト評価（SBOM 手動）.....	124
5.4.3.1.1.	初期コスト（SBOM 手動）.....	124
5.4.3.1.2.	運用コスト（SBOM 手動）.....	125
5.4.3.2.	コスト評価（SBOM 自動）.....	126

5.4.3.2.1. 初期コスト（SBOM 自動）	126
5.4.3.2.2. 運用コスト（SBOM 自動）	127
5.4.3.3. コスト評価（SBOM 自動・有償ツール）	134
5.4.3.3.1. 初期コスト（SBOM 自動・有償ツール）	134
5.4.3.3.2. 運用コスト（SBOM 自動・有償ツール）	134
5.4.4. 効果比較評価	141
5.4.4.1. コスト低減効果	141
5.4.4.2. 脆弱性リスク低減効果	145
5.4.4.3. ライセンス管理効果	146
5.4.5. 部品管理の比較評価の留意点・制約事項	147
5.5. 実証成果と課題の整理	149
5.5.1. SBOM のメリット等	149
5.5.1.1. SBOM の費用・効果の評価とメリット	149
5.5.1.2. SBOM の有効性	151
5.5.2. 検討すべき課題	151
5.5.3. 今後の取組の方向性整理	152
6. ソフトウェアタスクフォースの運営	154
6.1. タスクフォースの実施概要	154
6.1.1. 第 5 回ソフトウェアタスクフォース	154
6.1.2. 第 6 回ソフトウェアタスクフォース	154
6.2. 実施スケジュール	155
6.3. タスクフォースの実施結果の要点整理	155
6.3.1. 第 5 回ソフトウェアタスクフォースの意見概要	155
6.3.2. 第 6 回ソフトウェアタスクフォースの意見概要	156
7. 今後の課題と取組案のまとめ	157
8. 付録	160
8.1. タスクフォース議事要旨	160
8.1.1. 第 5 回ソフトウェアタスクフォース	160
8.1.2. 第 6 回ソフトウェアタスクフォース	162

別紙 1 OSS の利活用及びそのセキュリティ確保に向けた管理手法に関する事例集(改訂版)

図目次

図 2-1	大統領令におけるソフトウェア・サプライチェーンに関するタイムライン	5
図 2-2	ソフトウェア検証において推奨される 11 の最低基準	12
図 2-3	DHS と調達契約を行うベンダーに対するソフトウェアに関する要求事項	13
図 2-4	SBOM の最小要素の定義	14
図 2-5	大統領令に対する NIST の取組概要と C-SCRM との対応関係	20
図 2-6	ソフトウェア・サプライチェーンのセキュリティ確保のための新たな 4 つの概念	20
図 2-7	連邦政府機関がセキュアなソフトウェアを調達するための 4 つの勧告事項	24
図 2-8	sigstore のアーキテクチャ	49
図 2-9	英国国家サイバー戦略の 5 つの柱の概要	53
図 2-10	フランスの公的機関による OSS レポジトリを集めた Web サイトの概要	54
図 3-1	調査の観点とヒアリング調査等の項目との関係	61
図 4-1	本章の調査観点とヒアリング調査等の項目との関係	72
図 4-2	NIST 公式サイト上の OpenSCAP 実装の認証状況	81
図 4-3	ユーザーのインストール及びメンテナンスのモデル	85
図 4-4	Wordpress に関連する CVE 発行数の統計	90
図 4-5	JPCERT/CC からみた脆弱性ハンドリング (JPCERT/CC サイトより)	92
図 4-6	CNA の構造 (MITRE Corporation より)	95
図 4-7	OSV のワークフロー (出典: Google Open Source Blog / Launching OSV - Better vulnerability triage for open source)	100
図 4-8	米国政府脆弱性情報管理の概略 (案)	103
図 5-1	SBOM の有効性に関する比較評価の考え方の整理	107
図 5-2	実証におけるソフトウェア・サプライチェーンの構成	109
図 5-3	実証項目の基本構成の考え方	109
図 5-4	実証における比較評価シナリオ	114
図 5-5	実証の実施スケジュール	115

表 目 次

表 2-1	米国大統領令に関連する取組動向の調査対象.....	2
表 2-2	「重要なソフトウェア」の対象となる暫定ソフトウェアカテゴリ	6
表 2-3	「重要なソフトウェア」のセキュリティ対策の 5 つの目的と対策例.....	9
表 2-4	SBOM 最小要素における「データフィールド」の定義	14
表 2-5	SBOM を運用するために考慮すべき要件やプラクティス	15
表 2-6	「重要なソフトウェア」のセキュリティ対策実装に向けた覚書における要求事項	19
表 2-7	セキュアなソフトウェアを開発するための手法に関するフレームワーク (SSDF)	21
表 2-8	NTIA の Software Component Transparency に関連する取組動向の調査対象.....	26
表 2-9	NTIA による SBOM ツールの分類.....	27
表 2-10	INL によるエネルギー分野の SBOM PoC の概要.....	28
表 2-11	SBOM に関する誤解と事実	35
表 2-12	SBOM-a-rama の 2 日目の議論により抽出された主要な課題（抜粋）	42
表 2-13	国内外における医療機器分野の SBOM に関する取組概要	45
表 2-14	「Security Scorecards 2.0」の評価項目	49
表 2-15	ENISA により提案されたサプライチェーン攻撃の分類法.....	52
表 2-16	ソフトウェアに関連するインシデント・脆弱性事例.....	56
表 3-1	公開文献調査概要	62
表 3-2	公開文献事例 1 概要	63
表 3-3	公開文献事例 2 概要	63
表 3-4	公開文献事例 3 概要	64
表 3-5	公開文献事例 4 概要	65
表 3-6	公開文献事例 5 概要	65
表 3-7	公開文献事例 6 概要	66
表 3-8	公開文献事例 7 概要	67
表 3-9	公開文献事例 8 概要	67
表 3-10	ヒアリング調査概要	69
表 4-1	OpenSCAP サイト掲載のオープンソースの意義.....	80
表 4-2	OpenSCAP の採用状況.....	80
表 4-3	SCAP Security Guide の主要リスト	82
表 4-4	配布される方法の種類とアップデートの流れ.....	85
表 4-5	パッケージ管理コマンドと採用しているディストリビューション.....	86
表 4-6	OpenChain ISO/IEC 5230:20203.1.1 と 3.1.5 の相違点	87
表 4-7	Root CNA 一覧 (2022/01/05 時点).....	89
表 4-8	脆弱性対応の想定ケース	94
表 4-9	CVE プログラム参加組織数	96
表 4-10	CVE プログラム参加組織の種類	96
表 4-11	CVE プログラム参加組織の国別（上位 8 カ国）	96

表 4-12	関連する組織例	97
表 4-13	OSV のコンセプト概要	99
表 4-14	バグバウンティプログラム例 (Microsoft Corporation)	101
表 4-15	バグバウンティプログラム例 (Meta Platforms, Inc.)	101
表 4-16	バグバウンティプログラム例 (Google LLC)	102
表 4-17	バグバウンティプログラム例 (サイボウズ株式会社)	102
表 4-18	バグバウンティプログラム例 (LINE 株式会社)	102
表 5-1	SBOM 導入・活用に関する主な課題・障害	105
表 5-2	対象分野及びソフトウェア	108
表 5-3	SBOM 実証の評価項目及び比較評価方法に関する検討	111
表 5-4	実証におけるコスト測定手順	116
表 5-5	実証における効果測定手順	120
表 5-6	初期コストの主な内容 (独自形式)	122
表 5-7	部品情報生成・共有コストの主な内容 (独自形式)	123
表 5-8	初期コストの主な内容 (SBOM 手動)	124
表 5-9	初期コストの主な内容 (SBOM 自動)	126
表 5-10	無償ツールの試行において発生した障害・対応結果・得られたノウハウ	126
表 5-11	SPDX 項目に対する Syft 生成 SBOM での対応状況	128
表 5-12	実証で作成した SPDX 準拠の SBOM における SPDX 項目の有無	132
表 5-13	各ツールの SBOM における対応項目名	132
表 5-14	Black Duck 生成 SBOM での SPDX 対応状況	134
表 5-15	Black Duck が生成した SBOM における大統領令最小要件の対応状況	140
表 5-16	コスト低減効果比較	141
表 5-17	SBOM 作成コストの比較	143
表 5-18	SBOM 活用コストの比較	144
表 5-19	リスク低減効果の比較	145
表 5-20	部品管理対象件数・脆弱性発見数の比較	146
表 5-21	脆弱性特定コストの低減効果の評価	146
表 5-22	ライセンス違反特定コストの低減効果の測定	147
表 5-23	環境整備において発生した障害・対応内容・得られたノウハウ	149
表 5-24	SBOM 実証シナリオにおける主な情報の整理	150
表 7-1	今後の課題	158
表 7-2	今後の課題と取組策の対応関係	159

用語集・略語集

本報告書では、以下のとおり用語を定義する。

用語	概要
CISA	Cybersecurity and Infrastructure Security Agency の略で、米国国土安全保障省に属するサイバーセキュリティ・インフラストラクチャ・セキュリティ庁のこと。
CPE	Common Platform Enumeration の略で、情報システムを構成する、ハードウェア、ソフトウェアなどを識別するための共通の名称基準のこと。MITRE が中心となり仕様が策定された。
CSIRT	Computer Security Incident Response Team の略で、セキュリティインシデントに関する報告を受け、調査や対応活動を行う組織体のこと。
CVE	Common Vulnerabilities and Exposures の略。情報セキュリティにおける脆弱性やインシデントに関する識別子で、脆弱性に一つの CVE ID が付与される。米国非営利団体 MITRE が採番。
CVSS	Common Vulnerability Scoring System の略。脆弱性の深刻度を同一の基準の下で定量的に比較できる評価方法であり、0～10.0 の間でスコアが定まる。FIRST (Forum of Incident Response and Security Teams) が管理。
CycloneDX	SBOM の規格の一つで、OWASP (Open Web Application Security Project) が中心となって開発された。
DINUM	Direction Interministérielle du Numérique の略で、フランスのデジタル省庁間総局のこと。
DHS	Department of Homeland Security の略で、米国国土安全保障省のこと。
DISA	Defense Information System Agency の略で、米国国防総省の内局である米国国防情報システム局のこと。
DoC	Department of Commerce の略で、米国商務省のこと。
ENISA	European Network and Information Security Agency の略で、欧州連合の機関である欧州ネットワーク情報セキュリティ機関のこと。
FAR	Federal Acquisition Regulations の略で、米国の政府調達に関する一般的な調達原則を定めた連邦調達規則のこと。
FDA	Food and Drug Administration の略で、米国食品医薬品局のこと。
IMDRF	International Medical Device Regulators Forum の略で、国際的な医療機器規制の整合化と収束を促進する世界各国の医療機器規制当局による任意の活動のこと。日本語では国際医療機器規制当局フォーラムと呼ばれる。
INL	Idaho National Laboratory の略で、米国エネルギー省の国立研究所の一つであるアイダホ国立研究所のこと。
JVN	Japan Vulnerability Notes の略で、日本で流通・利用されているソフトウェアの脆弱性に関連する情報を収集・公開するデータベースのこと。JPCERT/CC と IPA により共同で運営されている。

用語	概要
MITRE ATT&CK	戦術や手法にまとめられ、少数のマトリックスだけでなく STIX/TAXII でも表現された既知の攻撃者の行動の構造化されたリストのこと。
NAS	Network Attached Storage の略で、ネットワーク上に接続可能な記憶装置のこと。
NIST	National Institute of Standards and Technology の略で、米国国立標準技術研究所のこと。
NIST NVD	NIST が管理している脆弱性情報データベース（NVD：National Vulnerability Database）のこと。
NTIA	National Telecommunications and Information Administration の略で、米国商務省電気通信情報局のこと。
NSA	National Security Agency の略で、米国国家安全保障局のこと。
OMB	Office of Management and Budget の略で、大統領行政府の機関の一つである米国行政管理予算局のこと。
PSIRT	Product Security Incident Response Team の略で、自社で製造・開発する製品やサービスを対象に、セキュリティレベルの向上やインシデント発生時の対応を行う組織のこと。
SAN	Storage Area Network の略で、複数のコンピュータと記憶装置を結ぶネットワークのこと。
SAST	Static Application Security Testing の略で、アプリケーションのソースコードやバイナリコード等を静的に解析し、脆弱性を特定する解析手法のこと。
SCA	Software Composition Analysis の略で、ソフトウェアが使用している部品（特に OSS 部品）の情報を明らかにする分析手法のこと。
SIEM	Security Information and Event Management の略で、ネットワーク製品やセキュリティ製品を含む機器のログを一元管理・解析し、インシデントにつながる脅威を検知するセキュリティ製品のこと。
SPDX	Software Package Data Exchange の略で、SBOM の規格の一つ。Linux Foundation が中心となって開発され、ISO/IEC 5962:2021 として国際標準化されている。
SWID	Software ID の略で、SBOM の規格の一つ。ISO/IEC 19770-2 として国際標準化されているほか、NISTIR 8060 として、NIST より SWID 作成のガイドラインが発表されている。

1. 背景と目的

産業活動のサービス化に伴い、産業に占めるソフトウェアの重要性は高まる傾向にある。特に、近年は、産業機械や自動車などの制御にもソフトウェアの導入が進んでおり、IoT 機器・サービスや今後普及が見込まれる 5G 技術においても、汎用的な機器でハードウェア・システムを構築したうえで、ソフトウェアにより多様な機能を持たせることで、様々な付加価値を創出していくことが期待される。

ソフトウェアを利活用した製品・サービスの安全・安心を担保するには、利活用するソフトウェアの脆弱性の管理が求められる。セキュリティ・バイ・デザインの考え方に基づいて、企画・設計段階で脆弱性を含めないようソフトウェアが構成されていたとしても、リリース後に脆弱性が発見されることが多く、ソフトウェアを利活用する側でのソフトウェア更新等の対応が求められる。また、自社の製品・サービスで利活用しているソフトウェアの保守・サポートが終了する場合には、それ以降に発見された脆弱性の管理について代替ソフトウェアへの変更を含めた検討が求められる。しかしながら、オープンソース・ソフトウェア（OSS）の利用が一般化する中で、自社製品において利用するソフトウェアであっても、コンポーネントとしてどのようなソフトウェアが含まれているのかを把握することが困難な状況が生じており、脆弱性の管理を事業者単独で実施することは費用対効果の面からも困難である。

このような課題を踏まえ、米国商務省の電気通信情報局（NTIA）では、平成 30 年 7 月から Software Component Transparency という官民合同の検討体制を構築し、ソフトウェアの成分構成を表す Software Bill of Materials（SBOM）をサプライチェーンで共有すること等を通じたソフトウェアの透明性向上に向けた実証実験を、ヘルスケア産業をはじめ複数の産業分野において行っている。

経済産業省では、「サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース（以下、「ソフトウェアタスクフォース」という。）」を令和元年 9 月に設置し、SBOM も含めたソフトウェア管理手法等に関して幅広い議論を行っているが、ソフトウェア分野における業界構造や商習慣が異なる我が国においては、総論として SBOM の有用性は理解されているものの、実際の活用に向けては様々なハードルがあると認識している。

本事業は、ソフトウェアを安全に利活用するための脆弱性の管理の在り方や、安全な OSS の選定及び活用するための枠組みを検討するために、OSS の利活用に起因するセキュリティリスクを洗い出すとともに、国内外におけるソフトウェアの脆弱性の管理の取組動向や、OSS を含むソフトウェアを企業等で自社製品・サービスとして活用する際のセキュリティリスクに対応するための取組について調査を行い、多角的な観点から OSS を含むソフトウェアを安全に利活用できる基盤の構築に向けた検討の基礎資料を作成し、我が国のサイバーセキュリティ政策立案に資することを目的とする。併せて、我が国の企業やサプライチェーンにおける SBOM の導入・活用に関する調査及び実証事業を通じ、我が国における SBOM の普及への課題や効果等について整理を行うものである。

2. 産業活動においてソフトウェアを安全に利活用するための枠組みに関する取組動向等の調査

本項目では、企業等において自社製品・サービスの一部として利活用されるソフトウェアのセキュリティ確保を実現するためのソフトウェア管理に関連する国内外の取組について調査した。特に、以下の項目について調査等を実施した。

1. NTIA 等の米国政府機関における取組動向の調査
2. 企業や業界団体、OSS コミュニティにおけるソフトウェア管理の取組動向の調査
3. OSS を含むソフトウェアの安全な利活用に関する基礎調査
4. 米国 CISA との勉強会の開催

2.1. NTIA 等の米国政府機関における取組動向の調査

本項目では、2021 年 5 月の米国大統領令を受けた米国政府機関の動向や、商務省電気通信情報局（NTIA）で実施されている Software Component Transparency の取組に関連する動向について調査した。

2.1.1. 米国大統領令に関連する取組動向の調査

2021 年 5 月 12 日、バイデン米国大統領は、サイバーセキュリティ強化のための大統領令「Executive Order on Improving the Nation's Cybersecurity (EO 14028)」に署名した。本大統領令では、政府機関に提供されるソフトウェアのサプライチェーンセキュリティの確保に関する条項も含まれ、大統領令発出後、複数の政府機関において様々な取組が実施されてきた。本事業で調査した米国大統領令に関連する取組動向の調査対象の一覧を表 2-1 に示す。以降では、各取組の概要、その取組が与える影響等について示す。

表 2-1 米国大統領令に関連する取組動向の調査対象

#	取組名・文書名	発行日	発行主体
1-1	Executive Order on Improving the Nation's Cybersecurity (EO 14028) ¹	2021 年 5 月 12 日	White House
1-2	Definition of Critical Software Under Executive Order (EO) 14028 ²	2021 年 6 月 25 日	NIST
1-3	Security Measures for “EO-Critical Software” Use Under Executive Order (EO) 14028 ³	2021 年 7 月 9 日	NIST

¹ <https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/>

² <https://www.nist.gov/system/files/documents/2021/10/13/EO%20Critical%20FINAL.pdf>

³ <https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity/security-measures-eo-critical-software-use-2>

#	取組名・文書名	発行日	発行主体
1-4	Recommended Minimum Standards for Vendor or Developer Verification (Testing) of Software Under Executive Order (EO) 14028 ⁴	2021 年 7 月 9 日	NIST
1-5	H.R.4611 - DHS Software Supply Chain Risk Management Act of 2021 ⁵	2021 年 7 月 21 日	DHS
1-6	The Minimum Elements For a Software Bill of Materials (SBOM) ⁶	2021 年 7 月 12 日	NTIA
1-7	Protecting Critical Software Through Enhanced Security Measures ⁷	2021 年 8 月 10 日	OMB
1-8	SP 800-161 (Cybersecurity Supply Chain Risk Management (C-SCRM) Practices for Systems and Organizations) Rev. 1 (Draft) ⁸	2021 年 10 月 28 日	NIST
1-9	Readout of White House Meeting on Software Security ⁹	2021 年 1 月 13 日	White House
1-10	SP 800-218 Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities ¹⁰	2022 年 2 月 3 日	NIST
1-11	Recommended Criteria for Cybersecurity Labeling of Consumer Software ¹¹	2022 年 2 月 4 日	NIST
1-12	Software Supply Chain Security Guidance Under Executive Order (EO) 14028 Section 4e ¹²	2022 年 2 月 4 日	NIST
1-13	OMB Statement on “Enhancing The Security Of Federally Procured Software” ¹³	2022 年 3 月 9 日	OMB

⁴ <https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity/recommended-minimum-standards-vendor-or-developer-verification-testing-software-under-executive-order-14028>

⁵ <https://www.congress.gov/bills/117/congress/house/bills/4611>

⁶ <https://www.ntia.doc.gov/report/2021/minimum-elements-software-bill-materials-sbom>

⁷ <https://www.whitehouse.gov/wp-content/uploads/2021/08/M-21-30.pdf>

⁸ <https://csrc.nist.gov/publications/detail/sp/800-161/rev-1/draft>

⁹ <https://www.whitehouse.gov/briefing-room/statements-releases/2022/01/13/readout-of-white-house-meeting-on-software-security/>

¹⁰ <https://csrc.nist.gov/publications/detail/sp/800-218/final>

¹¹ <https://csrc.nist.gov/publications/detail/white-paper/2022/02/04/recommended-criteria-cybersecurity-labeling-of-consumer-software/final>

¹² <https://csrc.nist.gov/publications/detail/white-paper/2022/02/04/software-supply-chain-security-guidance-eo-14028-section-4e/final>

¹³ <https://www.whitehouse.gov/omb/briefing-room/2022/03/07/omb-statement-on-enhancing-the-security-of-federally-procured-software/>

2.1.1.1. Executive Order on Improving the Nation's Cybersecurity (EO 14028)

2021年5月12日、バイデン米国大統領は連邦政府機関におけるサイバーセキュリティ改善に係る大統領令「Executive Order on Improving the Nation's Cybersecurity (EO 14028)」に署名した。本大統領令における主な指示事項は以下のとおりであり、官民での脅威情報の共有、ソフトウェアサプライチェーンセキュリティ対策の強化、ゼロトラストアーキテクチャへの移行等を通じて、連邦政府機関のサイバーセキュリティ対応能力の向上を図っている。

1. 官民の脅威情報共有における障害の除去（大統領令 Sec. 2）：

IT サービスプロバイダーが連邦政府と確実に脅威情報を共有できるようにしたうえで、特定のインシデント情報の共有を義務づける。

2. 連邦政府におけるより強力な標準の近代化・導入（大統領令 Sec. 3）：

FedRAMP の改定等を通じて、連邦政府が安全なクラウド及びゼロトラストアーキテクチャに移行することを支援し、多要素認証と暗号化の導入を義務づける。

3. ソフトウェア・サプライチェーンのセキュリティ向上（大統領令 Sec. 4）：

NIST を通じて政府が調達するソフトウェアの開発に関するセキュリティ基準（安全な開発環境の確保や構成要素に関する詳細（SBOM）の開示等を含む）を確立し、特に重要なソフトウェアに対して一定の対策を義務づける。また、DoC は、既存のラベル表示などを参考にして、消費者向けの情報提供に関するパイロット制度を開始する。

4. サイバー安全審査委員会の創設（大統領令 Sec. 5）：

DHS は、重大なインシデントが生じた際に政府と民間事業者が共同議長を務める「サイバー安全審査委員会」を設置し、サイバーセキュリティ向上に向けた具体的な提言を行う権限を与える。

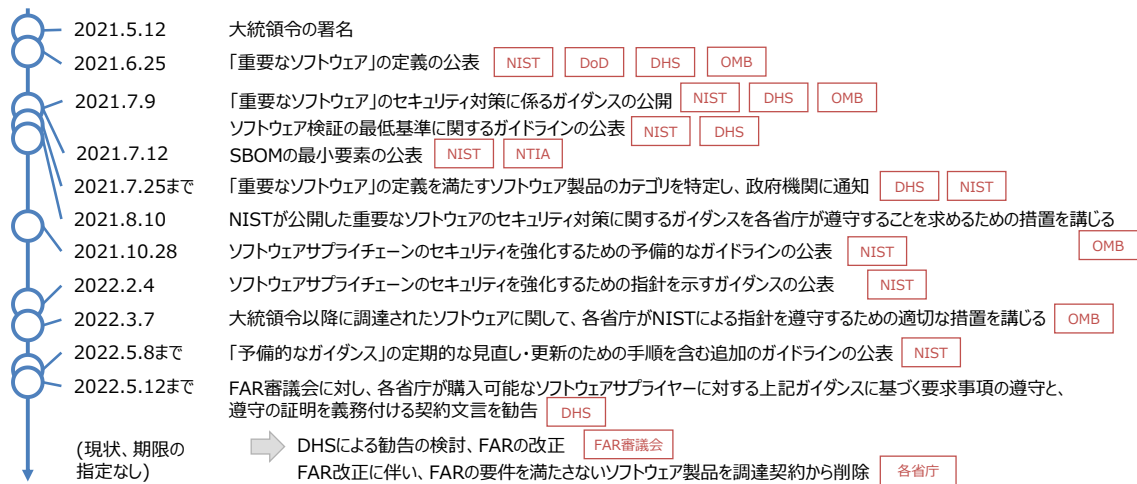
5. インシデント対応のための標準プレイブックの策定（大統領令 Sec. 6, 7）：

DHS は、連邦政府機関によるインシデント対応のためのプレイブックを策定する。また、連邦政府機関は、エンドポイント検知・対応（EDR）イニシアチブを展開し、インシデントの検知、積極的なサイバーハンティング、有事対応をサポートする。

6. 調査及び修復能力の向上（大統領令 Sec. 8）：

連邦政府機関に対してセキュリティイベントログの要件を設け、侵入を検知し、対処する組織能力の向上を支援する。

本調査では、上記指示事項のうち「3. ソフトウェア・サプライチェーンのセキュリティ向上」に関連する連邦政府機関の動向に関して主に調査した。大統領令におけるソフトウェア・サプライチェーンに関するタイムラインは図 2-1 に示すとおりであり、大統領令では、ソフトウェア・サプライチェーンの確保に向け、NIST が中心となりガイドラインを策定する旨を指示した。なお、このガイドラインには製品購入者に対する SBOM 提供に関する項目も含まれる。加えて、NIST に対して、NTIA と連携して SBOM の最小要素を公表することを指示した。将来的には、公開されたソフトウェア・サプライチェーンに関するガイダンスの要求事項に基づき、連邦政府のソフトウェア調達に関する FAR（連邦調達規則）が改正される予定である。



出所) 大統領令及び関連する動向における公開情報に基づき三菱総合研究所作成

図 2-1 大統領令におけるソフトウェア・サプライチェーンに関するタイムライン

2.1.1.2. Definition of Critical Software Under Executive Order (EO) 14028

2021年6月25日、米国NISTは、大統領令を受け「重要なソフトウェア(Critical Software)」の定義に関する文書「Definition of Critical Software Under Executive Order (EO) 14028」を公開した。文書では、「重要なソフトウェア」に関する以下の5つの属性が定義され、いずれかの属性を1つ以上持つコンポーネントを有している、又は、直接的にその属性を有しているソフトウェアを「重要なソフトウェア」と定義している。

1. 権限昇格機能の実行や、権限管理に関する機能が設計されている。
(該当する製品種別例) ネットワーク管理システム、ネットワーク構成管理ツール、ネットワークトラフィック監視ツール 等
2. ネットワークやコンピュータリソースへ直接アクセスするか、アクセスする権限を有する。
(該当する製品種別例) Web ブラウザ、ルーティングプロトコル、DNS リゾルバやDNS サーバー、SDN 制御プロトコル、VPN ソフトウェア 等
3. データ又はOTへのアクセスを管理するように設計されている。
(該当する製品種別例) ID 管理システム、バックアップサービスシステム、リカバリマネージャー、NAS、SAN 等
4. ネットワーク制御、エンドポイントセキュリティ等におけるセキュリティ機能のような、信頼性が不可欠な機能を実行する。
(該当する製品種別例) OS、ハードディスク暗号化ソフトウェア、パスワードマネージャー、EDR、ファイアウォール、IDS/IPS 等
5. 特権アクセスにより、セキュリティ対策が行われている信頼された境界の外で動作する。
(該当する製品種別例) SIEM、リモート型脆弱性スキャンツール、パッチ管理ツール、アプリケーション構成管理ツール 等

なお、適用対象となるソフトウェアに関して、本番システム用に購入又は導入され、運用目的で使用されるすべての形式(スタンドアロンソフトウェア、クラウドベースのソフトウ

ウェア等)に適用される。そのため、調査やテストのみに使用されるソフトウェア等、実稼働している本番システムに導入されていないものは適用対象外となる。文書では、この「重要なソフトウェア」の定義に加え、「重要なソフトウェア」に分類されるソフトウェアカテゴリとして、11の暫定カテゴリが定義されている。この暫定カテゴリは表 2-2 に示すとおりであり、セキュリティ上重要な機能を備えている、もしくは攻撃された場合に重大な危害の可能性のあるスタンドアロンやオンプレミスのソフトウェアに焦点が当てられている。

表 2-2 「重要なソフトウェア」の対象となる暫定ソフトウェアカテゴリ

暫定ソフトウェア カテゴリ	説明	製品種別例
ID、識別情報、アクセス管理 (ICAM)	組織のユーザー、システム、デバイスのアクセス権を一元的に識別、認証、管理、又は強制的に決定するソフトウェア。	● ID 管理システム ● ID プロバイダー及びフェデレーションサービス ● 証明書発行者 ● クラウドアクセスセキュリティブローカー (CASB) ● アクセス権限管理ソフトウェア ● 公開鍵基盤 (PKI)
OS、ハイパーバイザー、コンテナ環境	ハードウェアリソース (ベアメタル又は仮想化・コンテナ化を含む) のアクセスと制御を確立し、アプリケーションやインタラクティブユーザーにアクセス管理、メモリ管理、ランタイム実行環境などのサービスを提供するソフトウェア。	● サーバー、デスクトップ、及びモバイルデバイス用の OS ● OS 及び類似の環境の仮想化を実現するハイパーバイザー及びコンテナランタイムシステム
Web ブラウザ	ネットワークを介して Web サーバーによって配信されるコンテンツを処理し、デバイス及びサービス設定機能へのユーザーインターフェースとして使用されるソフトウェア。	● スタンドアロンブラウザと埋め込みブラウザ

暫定ソフトウェア カテゴリ	説明	製品種別例
エンドポイントセキュリティ	エンドポイントにインストールされたソフトウェア。通常は、エンドポイントのセキュアな操作を可能にしたり、その操作に貢献したり、又はエンドポイントに関する情報の詳細な収集を可能にする管理者特権を有する。	● ハードディスク暗号 ● パスワードマネージャー ● 悪意のあるソフトウェアを検索、削除、又は隔離するソフトウェア ● エンドポイントのセキュリティ状態（脆弱性及び設定）を報告するソフトウェア ● ファームウェア、OS、アプリケーション、ユーザー及びサービスアカウント、ランタイム環境の状態に関する詳細情報を収集するソフトウェア
ネットワーク制御	ネットワーク全体のデータフローを構成、制御、監視、保護するためのプロトコル、アルゴリズム、機能を有するソフトウェア。	● ルーティングプロトコル ● DNS リゾルバやDNS サーバー ● SDN 制御プロトコル ● VPN ソフトウェア ● DHCPなどの通信プロトコル
ネットワーク保護	悪意のあるネットワークトラフィックがネットワークセグメント又はシステム境界に出入りするのを防ぐ製品。	● ファイアウォール、IDS/IPS ● ポリシーベースのネットワーク管理システム（PBNM）
ネットワークの監視と構成管理	様々なシステムの状態を変更できる（又は、エージェントがインストールされている、特別な権限を有する）ネットワークベースの監視・構成管理ソフトウェア。	● ネットワーク管理システム ● ネットワーク構成管理ツール ● ネットワークトラフィック監視システム
運用の監視と分析	リモートシステムに関する運用状況やセキュリティ情報を報告するソフトウェア、及びその情報の処理、分析、応答に使用されるソフトウェア。	● SIEM

暫定ソフトウェア カテゴリ	説明	製品種別例
リモートスキャン	公開されたサービスのネットワークスキャンを実行することにより、ネットワーク上のエンドポイントの状態を検知するソフトウェア。	● 脆弱性を検出し、管理するソフトウェア
リモートアクセス と構成管理	リモートシステム管理や構成管理、又は他システムのリモート制御を行うソフトウェア。	● ポリシー管理 ● アップデート、パッチ管理 ● アプリケーション構成管理システム ● リモートアクセス・共有ソフトウェア ● 資産の検出と管理システム ● モバイルデバイス管理システム
バックアップ・リカバリとリモートストレージ	エンドポイント又は他のネットワークデバイスに保存されたコピーを作成し、転送するソフトウェア。	● バックアップサービスシステム ● リカバリマネージャー ● NAS、SAN

出所) NIST 文書に基づき三菱総合研究所作成

2.1.1.3. Security Measures for “EO-Critical Software” Use Under Executive Order (EO) 14028

2021 年 7 月 9 日、米国 NIST は、大統領令を受け「重要なソフトウェア」のセキュリティ対策を示すガイダンス「Security Measures for “EO-Critical Software” Use Under Executive Order (EO) 14028」を公開した。このガイダンスでは、表 2-3 に示すとおり「重要なソフトウェア」の使用にあたっての 5 つのセキュリティ対策の目的が定義され、それぞれの目的を達成するためのセキュリティ対策例が記載されている。また、セキュリティ対策例の実装を支援するために、参照することが望まれる NIST や CISA 等の既存のガイドラインも明記されている。

表 2-3 「重要なソフトウェア」のセキュリティ対策の5つの目的と対策例

「重要なソフトウェア」のセキュリティ対策の目的	セキュリティ対策例
重要なソフトウェアを実行するためのプラットフォーム¹⁴を、不正アクセスや不正利用から保護する。	● 重要なソフトウェアや実行するためのプラットフォームのすべてのユーザーと管理者に対して、多要素認証を使用する。 ● 重要なソフトウェアや実行するためのプラットフォームにアクセスしようとする各サービスを一意に識別及び認証する。 ● 重要なソフトウェアや実行するためのプラットフォームのネットワークベースの管理については、特権アクセス管理の原則に従う。 ● 重要なソフトウェアや実行するためのプラットフォーム、及び関連データへの直接のアクセスを最小限に抑えるために、必要に応じて境界保護技術を採用する。
重要なソフトウェアを実行するためのプラットフォームで使用するデータの機密性、完全性、可用性を保護する。	● 重要なソフトウェアや実行するためのプラットフォームのデータインベントリを管理する。 ● 重要なソフトウェアや実行するためのプラットフォームで使用するデータとリソースに対して、細かいアクセス制御を行い、可能な限り最小権限の原則に従う。 ● 重要なソフトウェアや実行するためのプラットフォームで使用する機密データを NIST の暗号化標準に準拠して暗号化することにより、保存データを保護する。 ● 可能な場合は、常に相互認証を使用し、NIST の暗号化標準に準拠した重要なソフトウェアや実行するためのプラットフォームの機密データ通信を暗号化することにより、転送中のデータを保護する。 ● 重要なソフトウェアや実行するためのプラットフォームで使用するデータをバックアップし、バックアップからいつでも復元できるように準備する。

¹⁴ エンドポイントの端末、サーバー、クラウドサービスのリソース等の「重要なソフトウェア」が動作するプラットフォームを意味する。

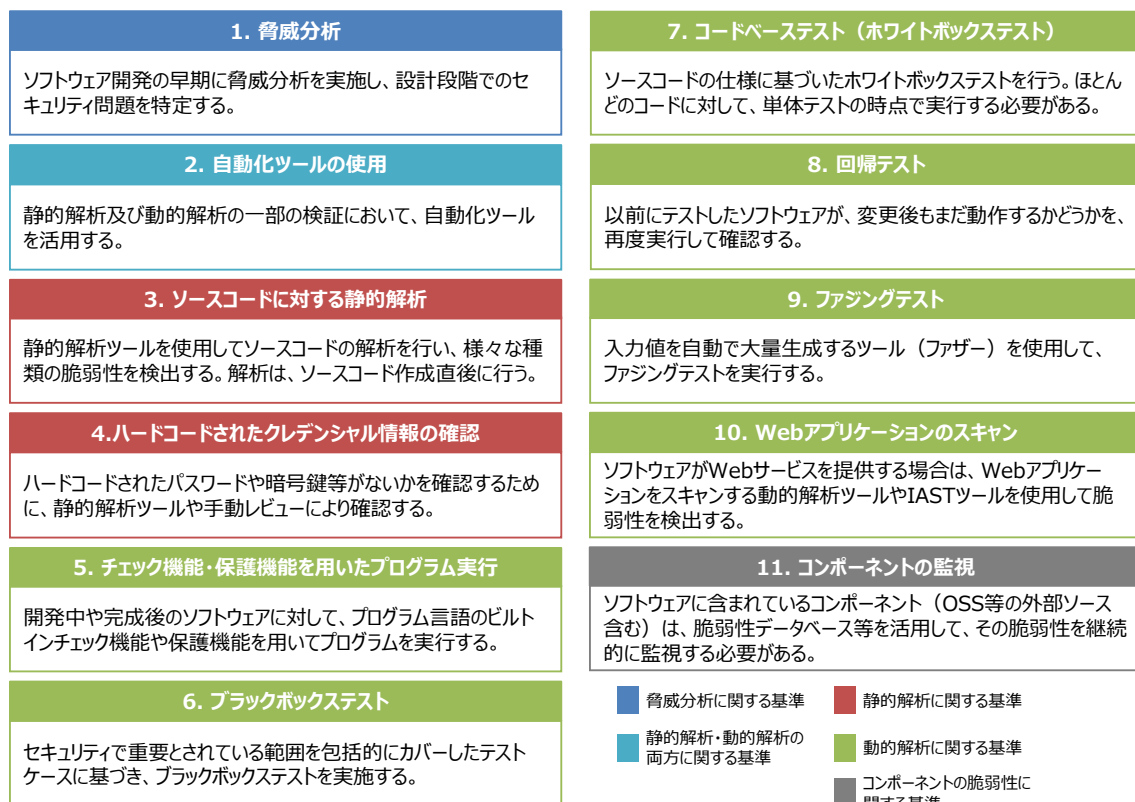
「重要なソフトウェア」のセキュリティ対策の目的	セキュリティ対策例
重要なソフトウェアを実行するためのプラットフォームと、それらのプラットフォームに展開されているすべてのソフトウェアを特定して管理し、重要なソフトウェアが悪用されることから保護する。	● 重要なソフトウェアや実行するためのプラットフォームと、各プラットフォームに展開されているすべてのソフトウェア（重要なソフトウェアとそうでないものの両方）を特定し、ソフトウェアインベントリを管理する。 ● 重要なソフトウェアを実行するためのプラットフォームとそれらのプラットフォームに展開されているすべてのソフトウェアを特定し、パッチ管理を行う。 ● 重要なソフトウェアを実行するためのプラットフォームとそれらのプラットフォームに展開されているすべてのソフトウェアを特定し、構成管理を行う。
重要なソフトウェアや、当該ソフトウェアを実行するためのプラットフォームに関連する脆弱性やインシデントを早急に検出、対応、回復する。	● 重要なソフトウェアを実行するためのプラットフォームで実行されているすべてのソフトウェアに関連するセキュリティイベントに関して必要な情報を記録する。 ● 重要なソフトウェアを実行するためのプラットフォームで実行されているすべてのソフトウェアを監視する。 ● 重要なソフトウェアを実行するためのプラットフォームで実行されているソフトウェアを保護するために、エンドポイントセキュリティを行う。 ● 重要なソフトウェアプラットフォームとの間のネットワークトラフィックを監視し、プラットフォームとそのソフトウェアを保護するために、ネットワークセキュリティを行う。 ● 重要なソフトウェアやプラットフォームに関連するインシデントを処理する方法について、すべてのセキュリティ運用担当者とインシデント対応チームのメンバーをトレーニングする。

「重要なソフトウェア」のセキュリティ対策の目的	セキュリティ対策例
重要なソフトウェアや、当該ソフトウェアを実行するためのプラットフォームに関するユーザー及び管理者のセキュリティの理解を促進する。	● 重要なソフトウェアや実行するためのプラットフォームを安全に使用するための方法について、すべてのユーザーに対してトレーニングを行う。 ● 重要なソフトウェアや実行するためのプラットフォームを安全に管理するための方法について、すべての管理者に対してトレーニングを行う。 ● 重要なソフトウェアや実行するためのプラットフォームのすべてのユーザーや管理者に対してトレーニングを行い、トレーニングの効果を強化するための活動を行う。

出所) NIST 文書に基づき三菱総合研究所作成

2.1.1.4. Recommended Minimum Standards for Vendor or Developer Verification (Testing) of Software Under Executive Order (EO) 14028

2021 年 7 月 9 日、米国 NIST は、大統領令を受けソフトウェア検証の最低基準に関するガイドライン「Recommended Minimum Standards for Vendor or Developer Verification (Testing) of Software Under Executive Order (EO) 14028」を公開した。ガイドラインでは、図 2-2 に示すベンダーや開発者によるソフトウェア検証の際に推奨される 11 の最低基準が示された。最低基準は実行可能なコンピュータプログラムすべてに推奨され、将来的にはソフトウェアベンダーや開発者に対する強制基準の基礎となりうることが明記されている。



出所) NIST 文書に基づき三菱総合研究所作成

図 2-2 ソフトウェア検証において推奨される 11 の最低基準

文書ではソフトウェア検証における最低基準の補足として、具体的な技術や検証にあたって留意すべき点も紹介されている。例えば、静的解析において SAST ツールを使用することで、バッファオーバーフロー、SQL インジェクション、組織のコーディング規約への違反等の問題を発見できる一方で、SAST ツールでは、認証の問題、アクセス制御の問題、暗号化の安全でない使用など、特定の種類の脆弱性を見つけるのは困難であることが明記されている。また、多くの SAST ツールでは誤検知・誤検出が含まれているほか、ツールによって検知された問題が、脆弱性であるか、アプリケーションで問題が軽減されているかを判断できない点も留意すべきとしている。そのため、SAST ツールのユーザーは、ツールが検知する問題を優先順位付けし、最も重要な問題の修正に注力するべきとしている。

また、異なる留意点として、ソフトウェアに含まれているコンポーネントについて、既知の脆弱性を確認することの重要性を説いている。この際、コードスキャンツールを使用して、含まれているコンポーネントを特定し、既知の脆弱性を確認することが推奨されている。文書では、以下のコードスキャンツールが例示されている。

- Black Duck
- Binary Analysis Tool (BAT)
- Contrast Assess
- FlexNet Code Insight
- FOSSA
- JFrog Xray
- OWASP Dependency-Check
- Snyk

- Sonatype IQ Server
- SourceClear
- WhiteHat Sentinel SCA
- WhiteSourceBolt

2.1.1.5. H.R.4611 - DHS Software Supply Chain Risk Management Act of 2021

2021 年 7 月 21 日、Ritchie Torres 下院議員により、DHS のソフトウェア・サプライチェーンマネジメントに関する法案「H.R.4611 - DHS Software Supply Chain Risk Management Act of 2021」が下院に提出された。本法案は 2021 年 10 月 20 日に下院を通過した。本法案では、図 2-3 に示すとおり、DHS と調達契約を行うベンダーに対して、ソフトウェア・サプライチェーンに関する要求事項が規定されており、本法案が成立した場合、DHS が IT 機器やサービス¹⁵を調達する際に、機器・サービスのベンダーに対して機器・サービスの SBOM、脆弱性情報及び脆弱性がある場合の対処方法を提示することが義務化される。また、機器・サービス納入後も、SBOM に変更が発生した際に変更内容を随時提示することが義務化される。その他、会計検査院に対して、本法案成立日から 1 年以内に本法案の実施状況のレビューや対象となる契約に関するサプライチェーン改善に向けた推奨事項等を報告書としてまとめ、DHS や国土安全保障委員会等に提出することを求めている。本法案は大統領令に基づき策定されたもので、成立した場合、成立日から 180 日後に有効となる。

新規・既存の調達契約における SBOM、脆弱性情報、脆弱性がある場合の対処方法の提示	・ SBOMの提示： 入札提案時に、使用予定のコンポーネントに関するSBOMを提示する必要がある。 (既存の契約の場合は、DHSからの要請に応じて使用中のSBOMを提示する必要がある。) ・ 脆弱性が無いことの提示： 使用予定（または使用中）のコンポーネントにおいて、最終製品やサービスのセキュリティに影響を与える既知の脆弱性や欠陥が存在しないことを認証し、その認証結果を提示する必要がある。 ・ 脆弱性がある場合の対処方法の提示： 使用予定（または使用中）のコンポーネントに脆弱性や欠陥が存在する場合は、脆弱性と欠陥の情報及び、それらを軽減・対処するための計画を提示する必要がある。
SBOMの更新内容の提示	・ SBOMの更新内容を提示： DHSへ提示したSBOMに変更が発生した場合、変更内容を随時提示する必要がある。

出所) H.R.4611 法案に基づき三菱総合研究所作成

図 2-3 DHS と調達契約を行うベンダーに対するソフトウェアに関する要求事項

2.1.1.6. The Minimum Elements For a Software Bill of Materials (SBOM)

2021 年 7 月 12 日、米国 NTIA は、大統領令を受け SBOM の最小要素の定義を公開した。本定義の公開に向け、NTIA はパブリックコメントを実施し、ソフトウェア関連の企業や専門家からの意見を抽出した。図 2-4 に示すとおり、SBOM の最小要素には、「データフィールド」、「自動化サポート」、「プラクティスとプロセス」の 3 つのカテゴリが含まれ、コンポーネントを一覧化した部品表に含まれる情報だけでなく、SBOM の利活用者が実施すべき

¹⁵ 「IT 機器やサービス」には、米国通信法で定義された「電気通信機器」及び「電気通信サービス」や、合衆国法典で定義された「情報システム」及び「情報技術」が含まれ、具体的な例として、NIST の定義に基づくと、ルーターやスイッチ等の通信機器、OS、ソフトウェア、ファームウェア、アプリケーションのほか、産業制御システム等が挙げられる。

事項も規定されている。文書では、各省庁のソフトウェアに関する取組が本定義に基づき実施されることが明記されている。

3つのカテゴリ	「最小要素」の概要	「最小要素」の具体的な定義
データフィールド (Data Fields)	各コンポーネントに関する 基本情報を明確化すること	以下の情報をSBOMに含めること。 - コンポーネントのサプライヤー名 - コンポーネント名 - コンポーネントのバージョン - その他の一意な識別子 - 依存関係 - SBOMデータの作成者 - タイムスタンプ
自動化サポート (Automation Support)	SBOMの自動生成や 可読性などの自動化を サポートすること	SBOMデータは機械判読可能かつ相互運用可能なフォーマットを用いて作成され、共有されること。現状では、国際的な議論を通じて策定された、SPDX、CycloneDX、SWIDタグを用いること。
プラクティスとプロセス (Practices and Processes)	SBOMの要求、生成、 利用に関する運用方法を 定義すること	SBOMを活用する組織は、以下の項目に関する運用方法を定めること。 - SBOM作成頻度 - SBOMの深さ - 既知の未知 - SBOMの共有 - アクセス管理 - 誤りの許容

出所) NTIA 文書に基づき三菱総合研究所作成

図 2-4 SBOM の最小要素の定義

SBOM 最小要素のうち「データフィールド」では、ソフトウェアコンポーネントの追跡及び保守の対象となる情報が定義されている。これらの情報は、ソフトウェアを構成するコンポーネントを理解するための情報を収集し、利用者に提示するといった、一貫性のある構造を確立するために必要不可欠となるためとしている。具体的なデータフィールドの一覧を表 2-4 に示す。データフィールドの活用目的として、コンポーネントに関する詳細情報の表現、ソフトウェア・サプライチェーン全体でのコンポーネントの追跡、脆弱性データベースやライセンスデータベースなどのデータソースとのマッピング等が挙げられている。

表 2-4 SBOM 最小要素における「データフィールド」の定義

項目	項目 (原文)	説明
コンポーネントのサプライヤー名	Supplier Name	コンポーネントを作成、定義、識別するエンティティの名称。
コンポーネント名	Component Name	サプライヤーによって定義されたソフトウェアのある単位に割り当てられた名称。
コンポーネントのバージョン	Version of the Component	過去に識別されたバージョンからのソフトウェアの変更を特定するために、サプライヤーが使用する識別子。
その他の一意の識別子	Other Unique Identifiers	コンポーネントを識別するために使用される、又は関連するデータベースの検索キーとして機能するその他の識別子。
依存関係	Dependency Relationship	コンポーネントがあるソフトウェアに含まれているという関係の特徴付ける。

項目	項目（原文）	説明
SBOM データの作成者	Author of SBOM Data	コンポーネントの SBOM データを作成するエンティティの名称。
タイムスタンプ	Timestamp	SBOM データを作成した日付と時刻の記録。

出所) NTIA 文書に基づき三菱総合研究所作成

SBOM 最小要素のうち、「自動化サポート」の位置づけとして、SBOM において自動生成や可読性などの自動化をサポートすることで、組織の境界を越えて SBOM の利活用を促進することが求められるとしている。現状では SBOM の利用に際して活用できる SPDX、CycloneDX、SWID タグ等のデータ形式が存在し、これらのデータ形式は、機械で読み取り可能であることに加えて、人間も可読であり、トラブルシューティングを適切にサポートするものであるとしている。さらに、これらの SBOM データ形式は最小要素のデータフィールドに対して相互運用可能であると見なされており、これらの形式のいずれかを用いて共有される必要があるとしている。

SBOM 最小要素のうち、「プラクティスとプロセス」に関して、SBOM を開発ライフサイクルに統合するために、SBOM を利活用する組織は、SBOM の仕組みに焦点を当てた特定のプラクティスとプロセスに従う必要があるとしている。具体的には、表 2-5 に示す項目を考慮し、SBOM を要求、生成、利用するためのポリシー、契約、ルール等では、いくつかの項目に対して明示的に対処する必要があるとしている。

表 2-5 SBOM を運用するために考慮すべき要件やプラクティス

項目	要件やプラクティス
SBOM 作成頻度	ソフトウェアコンポーネントが新たにビルド又は更新された場合、ソフトウェアの新しいバージョンを反映するために新たな SBOM を作成する必要がある。
SBOM の深さ	すべての依存関係がリスト化され、すべてのトップレベルのコンポーネントが明記されている必要がある。 少なくとも、すべてのトップレベルの依存関係は、依存関係を再帰的に探索するための詳細情報とともにリスト化される必要がある。
既知の未知 (Known Unknown)	完全な依存関係が SBOM に列挙されていない場合、SBOM 作成者は「既知の未知」が存在することを明示的に識別する必要がある。 すなわち、依存関係のデータでは、それ以上の依存関係がないコンポーネントと、依存関係の存在が不明な不完全なコンポーネントとを明確に区別する必要がある。
SBOM の共有 (配布と配信)	SBOM は、必要な人がタイムリーに利用できるようにする必要がある、適切なアクセス管理と権限設定がなされている必要がある。

項目	要件やプラクティス
アクセス管理	SBOM に対するアクセス管理を設定する。オープンソースのサプライヤーは、SBOM データを公開により自身の利益に役立つと考える場合があるが、他のサプライヤーは、SBOM データを自身で保持し、特定の顧客やユーザーへのアクセスを制限したいと考える可能性もある。
誤りの許容	ソフトウェアサプライチェーンデータの管理方法は日々進化し続けている。そのため、SBOM を導入し、運用する最初フェーズでは完全性が欠如する可能性があるが、SBOM の利用者は偶発的な誤りに対しては明確に許容すべきである。これにより、ツールの継続的な改善が促進される。

出所) NTIA 文書に基づき三菱総合研究所作成

文書では、SBOM の「最小要素」は、基本的なユースケースを実現するための諸段階のステップと要件であると位置づけており、ソフトウェア・サプライチェーンの透明性を拡大し、ソフトウェアを保護するための可視性を実現するためには、「最小要素」に加えて、以降に示す7つのテーマを実施・留意する必要があるとしている。

2.1.1.6.1. 追加で推奨されるデータフィールド

SBOM の「最小要素」で挙げたデータフィールドに、次の項目も加える必要がある。なお、多くの場合は、これらの項目は明確に定義され既に実装されているため、実装されていない場合は追加する必要がある。

- **コンポーネントのハッシュ値**

ソフトウェアのコンポーネントを脆弱性データソースなどの関連するデータソースにマッピングするには、セキュリティ面で強固な識別子が重要で、暗号化されたハッシュが役立つ。

- **ソフトウェアのライフサイクルフェーズ**

ソフトウェアは、ビルド時、バイナリ解析ツール使用後など、様々なフェーズがあり、SBOM が作成されたフェーズで内容に違いが生じる可能性がある。そのため、SBOM がいつ、どのように記録されたかを管理する必要がある。

- **他のコンポーネントとの関係**

SBOM の最小要素では、あるコンポーネントがソフトウェアに含まれているという関係に留まっている。より詳細な関係として、コンポーネントが他の既知のコンポーネントと類似しているか、コンポーネントに変更を加えたものかを捉える必要がある。

- **ライセンス情報**

SBOM は各コンポーネントのライセンス情報を共有するのに役立つ。SBOM を受け取ったユーザーやベンダーは、法的リスクを回避しながら、ソフトウェアを別のアプリケーションのコンポーネントとして使用できるかどうかを判断する。

2.1.1.6.2. クラウドベースソフトウェアや SaaS の課題

最近のソフトウェアの多くはクラウドや SaaS で提供されている。この提供形態は、SBOM に関して課題を提示している。

- **ソフトウェアのリスクの把握と対処の責任者**

クラウドや SaaS のソフトウェアは、ユーザー側のインフラで実行されておらず、ユーザーの管理下に無いため、脆弱性とリスクを把握して対処する責任はクラウドソフトウェアベンダーにある。

- **ソフトウェア・サプライチェーンのリスク把握**

ソフトウェアベンダーは、開発するソフトウェアのサプライチェーンから提供されるメタデータだけでなく、外部アプリケーションのサードパーティから提供されるメタデータを管理する必要がある。多くのアプリケーションはサードパーティのサービスも利用しており、サードパーティのサービスに関する完全なメタデータを取得することは必要だが、組織間の実装のためにまだ標準化されておらず、十分に成熟していない。

2.1.1.6.3. SBOM の整合性や完全性の担保

SBOM 利用者により、SBOM データが改ざんされていないかが確認される場合がある。SBOM を提供・要求する際は、SBOM データに署名し、改ざん検知を確認するためのオプションを検討することが推奨される。SBOM データの完全性と信頼性を維持することは政府機関にとって優先事項であり、今後、SBOM データの署名を要求されることも想定される。

2.1.1.6.4. 脆弱性と SBOM の関係

セキュリティ対策上の SBOM 活用の目的は、ソフトウェア・サプライチェーンの脆弱性とリスクを特定することである。一部のソフトウェア開発者は、SBOM データ内に脆弱性情報を組み込む場合がある。この問題点は、SBOM データは基本的に静的であり、ある時点では脆弱性が無くても、時間の経過とともに新たなバグが発見され、脆弱性となる可能性があるという点である。したがって、SBOM データの脆弱性情報は、完全で最新であると認識されることはほとんどなく、将来的には SBOM データにおいて脆弱性データソースを参照できるようにする必要がある。つまり、脆弱性に関するデータは SBOM とは別のデータ構造で特定できるようにすることが推奨される。脆弱性データが組織間で共有されている場合は、脆弱性データと SBOM の両方を共有し、アクセス管理して運用していくことになる。

2.1.1.6.5. 依存関係における脆弱性とエクスプロイト可能性

コンポーネントのすべての脆弱性が依存するソフトウェアに対してリスクをもたらすわけではない。リスク有無を判断するためには、2つのステップが必要である。

- コンポーネントのサプライヤーは、脆弱性が特定のソフトウェアに影響を与えないという信頼できる根拠に基づき判断する。
- 脆弱性を持つコンポーネントを含むソフトウェアが組織に影響を与えないと判断する。これは、脆弱性とその状態を関係付けるもので、Vulnerability Exploitability

eXchange (VEX) と呼ばれる。VEX により、特定のソフトウェアコンポーネントが脆弱性の影響を受けるかどうかを判断できる。SBOM データを分析するツールには、VEX データを自動的に反映する機能を組み込むことが推奨される。

2.1.1.6.6. レガシーソフトウェアのバイナリ解析

SBOM データは、効率性の観点からサプライヤーから提供される必要がある。ただし、状況によっては、SBOM 作成に使用できる情報はオブジェクトコードのみであり、ソースコードを使用できない場合もある。メンテナンスされていないソフトウェアは悪用されるリスクが高くなり、古いソフトウェアはメンテナンスされないリスクが高くなる。レガシーソフトウェアに対する既知の脆弱性によるリスクを評価する際は、レガシーソフトウェアの透明性を高めて理解する必要がある。このような場合、バイナリ解析ツールを使用することで、システムコンポーネントの依存関係をよりよく理解できる。バイナリ解析を使用して、SBOM の内容を検証するほか、SBOM データのギャップを理解することもできる。

2.1.1.6.7. SBOM の柔軟性と統一性

SBOM に限った問題ではないが、ソフトウェアにおけるセキュリティでは、柔軟性と統一性のバランスを取ることが困難である。どの組織においても、互換性が低い様々な SBOM を実装するためにはコストが必要となる。ソフトウェアのエコシステム全体で SBOM を正常に実装するには、幅広いルールとポリシーの両方に加えて柔軟性が必要となる。将来的には、SBOM の最小要素に基づいて構築されたすべての要件は、以下の 2 つの概念に基づき確立していく必要がある。

- すべてのセキュリティ対策、特に SBOM の利活用はプロセスであり、単一の目標ではない。
- SBOM の基本原則は透明性という効力であり、実行することに焦点を当てるべきである。

2.1.1.7. Protecting Critical Software Through Enhanced Security Measures

2021 年 8 月 10 日、OMB は、各省庁における「重要なソフトウェア」の対策実装に向けた覚書「Protecting Critical Software Through Enhanced Security Measures」に署名した。具体的な要求事項は表 2-6 に示すとおりであり、各省庁が利用・調達している「重要なソフトウェア」を 60 日以内に特定し、当該ソフトウェアに対して必要なセキュリティ対策を 1 年以内に実装することを指示したほか、各省庁のセキュリティ対策にあたって、NIST が 7 月 9 日に公開したガイダンスに遵守することを求めた。また、NIST や CISA に対して、「重要なソフトウェア」に含まれるソフトウェアカテゴリの定義や、当該ソフトウェアのセキュリティ対策に係るガイドラインを必要に応じて更新することを要求した。

表 2-6 「重要なソフトウェア」のセキュリティ対策実装に向けた覚書における要求事項

要求先	要件	期限
各省庁	使用中又は調達中のすべての「重要なソフトウェア」を特定する。	本覚書発行後、60 日以内
	■ 初期フェーズ セキュリティ上重要な機能を備えている、もしくは攻撃された場合に重大な危害の可能性のあるスタンドアロンやオンプレミスの「重要なソフトウェア」を対象に、NIST のガイダンスに記載されたセキュリティ対策 ¹⁶ を実装する。	本覚書発行後、1 年以内
	■ 後続フェーズ 今後、NIST が「重要なソフトウェア」のセキュリティ対策に係るガイダンスを更新した場合に、「重要なソフトウェア」に対して当該ガイダンスに記載された対策を実装する。	NIST によるガイダンスの更新後、1 年以内
	今後、CISA が「重要なソフトウェア」に新たなソフトウェアカテゴリを追加した場合に、NIST のガイダンスに記載されたセキュリティ対策を実装する。 【新たに対象となるソフトウェアの候補】 データへのアクセスを制御するソフトウェア、ソフトウェア開発ツール、OT のソフトウェアコンポーネント 等	CISA による「重要なソフトウェア」の定義更新後、1 年以内
NIST	「重要なソフトウェア」の定義と、当該ソフトウェアのセキュリティ対策に係るガイダンスを更新する。	必要に応じて
CISA	「重要なソフトウェア」の定義に含まれるソフトウェアカテゴリのリストを更新する。	必要に応じて

出所) OMB 文書に基づき三菱総合研究所作成

2.1.1.8. SP 800-161 (Cybersecurity Supply Chain Risk Management (C-SCRM) Practices for Systems and Organizations) Rev. 1 (Draft)

2021 年 10 月 28 日、米国 NIST は、SP 800-161 (Cybersecurity Supply Chain Risk Management (C-SCRM) Practices for Systems and Organizations) Rev. 1 のドラフト版を公開した。本文書の付録として、大統領令で指示されたソフトウェア・サプライチェーンのセキュリティ強化に向けた予備的なガイドラインが位置づけられている。本ガイドラインでは、図 2-5 に示すとおり、大統領令に対する NIST の取組と SP 800-161 Rev.1 で規定された C-SCRM の管理策との対応関係が整理されている。

¹⁶ NIST が 2021 年 7 月 9 日に公開した「重要なソフトウェア」のセキュリティ対策を示すガイダンス（第 2.1.1.3 項参照）に記載されたセキュリティ対策を意味する。

「重要なソフトウェア」の定義と求められる対策	- 各政府機関が優先的にセキュリティ対応を行う対象となる「重要なソフトウェア」の定義及び「重要なソフトウェア」のセキュリティ対策に関する取組。 C-SCRMの管理策や補足ガイダンスは、「重要なソフトウェア」に求められるセキュリティ対策を実施するための有効な手段である。
ソフトウェア検証の最低基準	- サプライヤー（ベンダーや開発者など）によるソフトウェア検証の際に推奨される最低基準に関する取組。 C-SCRMの管理策や補足ガイダンスを用いて、最低基準を満たしたソフトウェア検証が実施されていることを確認できる。
消費者向けIoTやソフトウェアのサイバーセキュリティラベリングプログラム	- 消費者向けのIoTデバイスやソフトウェアに対するラベリングのための仕組み構築に向けた取組。 - この取組はまだ検討段階にあるが、C-SCRMには、この検討に関する直接的な内容が記載されている。

出所) NIST 文書に基づき三菱総合研究所作成

図 2-5 大統領令に対する NIST の取組概要と C-SCRM との対応関係

加えて、文書ではソフトウェア・サプライチェーンのセキュリティ確保のための新たな 4 つの概念として、図 2-6 に示す概念を明記しており、それぞれの事項を実施する際に各省庁が留意すべき点、各概念を採用することによる効果、今後の発展可能性を概説している。

SBOM	各省庁は、ソフトウェア製品やサービスのサプライヤーが、大統領令及びNTIAが公表しているガイドラインに準拠したSBOMを作成していることを、確認すべきである。
サプライヤーにおけるリスク評価の強化	サプライヤーが作成したソフトウェアだけではなく、サプライチェーンにおいてソフトウェアに関与する事業体に対してもリスク評価を強化すべきである。
OSSの管理	各省庁は、サプライヤーのソフトウェアにおけるOSSの使用状況をより深く理解するよう努めるべきである。
脆弱性管理	ゼロトラストアーキテクチャの議論を踏まえ、各省庁の戦略は、検出された脆弱性をいかに効率的かつ包括的に管理するかについて焦点を当てるべきである。

出所) NIST 文書に基づき三菱総合研究所作成

図 2-6 ソフトウェア・サプライチェーンのセキュリティ確保のための新たな 4 つの概念

2.1.1.9. Readout of White House Meeting on Software Security

2022 年 1 月 13 日、米国ホワイトハウスはソフトウェアセキュリティに関する官民会議を開催した。会議には、国家安全保障サイバーセキュリティ担当官や DoD、DoC、CISA 等の政府機関のほか、Akamai、Amazon、Apple、Cloudflare、Facebook/Meta、GitHub、Google、IBM、Microsoft、Oracle、Red Hat、VMware 等の民間企業、Apache Software Foundation、Linux Foundation 等のソフトウェアに関する非営利団体も参加した。この会議は、2021 年 12 月に

発見された Log4j の脆弱性（表 2-16 参照）を踏まえ開催されたもので、オープンソース・ソフトウェアにおける脆弱性への対応策や、脆弱性の検出・修正のプロセス改善策、パッチプロセスの迅速化の方法等が議論された。SBOM に関する議論もなされたとしており、具体的には、購入・使用するソフトウェアの構成要素を容易に把握するために求められる SBOM 利活用の促進方法について議論がなされた。

2.1.1.10. SP 800-218 Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities

2022 年 2 月 3 日、米国 NIST はソフトウェアの脆弱性を軽減するためのソフトウェア開発者向けの手法をまとめたフレームワークである SSDF（Secure Software Development Framework）の Ver. 1.1 を SP 800-218 として公開した。表 2-7 に示すとおり、各手法は 4 つに分類され、手法を実践するためのタスクが体系化されている。各手法を実践することで、脆弱性を低減するとともに、未対処の脆弱性が悪用された場合の影響を軽減し、脆弱性の再発を防ぐ根本原因に対処可能としている。また、大統領令の記載事項と SSDF の手法との対応関係についても整理されており、大統領令の記載事項に対処するために SSDF を活用可能であるとしている。

表 2-7 セキュアなソフトウェアを開発するための手法に関するフレームワーク（SSDF）

分類	手法
1. 組織の準備 (PO) ソフトウェアを開発する組織は、組織レベルで安全なソフトウェアの開発を行うために、適した人材、プロセス、技術を準備する必要がある。	● ソフトウェア開発におけるセキュリティ要件を定義する (PO.1) ● ソフトウェア開発における役割と責任を明確化する (PO.2) ● ソフトウェア開発を支援するツールチェーンを明確化する (PO.3) ● ソフトウェアのセキュリティを確認するための基準を定義し、活用する (PO.4) ● ソフトウェア開発のための安全な環境を導入し、維持する (PO.5)
2. ソフトウェアの保護 (PS) ソフトウェアを開発する組織は、ソフトウェアのすべてのコンポーネントを、改ざんや不正アクセスから保護する必要がある。	● あらゆる形態のコードを不正アクセスや改ざんから保護する (PS.1) ● ソフトウェアリリースの完全性を検証する仕組みを提供する (PS.2) ● 各ソフトウェアのリリースをアーカイブ化し、保護する (PS.3)

分類	手法
3. 安全なソフトウェアの開発 (PW) ソフトウェアを開発する組織は、脆弱性を最小限に抑え、十分なソフトウェアを備えたソフトウェアをリリースする必要がある。	● セキュリティ要件を満足するとともにセキュリティリスクを軽減できるよう、ソフトウェアを設計する (PW.1) ● ソフトウェア設計をレビューし、セキュリティ要件やリスクへの適合性を検証する (PW.2) ● 実現可能な場合、機能を重複させずに既存の保護されたソフトウェアを再利用する (PW.4) ● セキュアコーディングのプラクティスを遵守してソースコードを作成する (PW.5) ● 実行可能なセキュリティを向上させるために、コンパイラ、インタプリタ及びビルドプロセスを構築する (PW.6) ● コードをレビュー・分析することで、脆弱性を特定し、セキュリティ要求事項への準拠を検証する (PW.7) ● 実行コードをテストして脆弱性を特定し、セキュリティ要求事項への準拠を検証する (PW.8) ● ソフトウェアをデフォルトで安全な設定とする (PW.9)
4. 脆弱性への対応 (RV) ソフトウェアを開発する組織は、リリースするソフトウェアに残存する脆弱性を特定し、適切に対応する必要がある。	● 脆弱性に対する継続的な把握と確認を実施する (RV.1) ● 脆弱性の評価、優先順位付け及び修正を実施する (RV.2) ● 脆弱性を分析することで、その根本原因を特定する (RV.3)

出所) NIST 文書に基づき三菱総合研究所作成

2.1.1.11. Recommended Criteria for Cybersecurity Labeling of Consumer Software

2022 年 2 月 4 日、米国 NIST は、大統領令に基づき消費者向けソフトウェアのセキュリティラベルの推奨基準「Recommended Criteria for Cybersecurity Labeling of Consumer Software」を公開した。具体的には、ベンダーがソフトウェアにラベリングするための技術的な基準、ラベリングの基準、ラベリングの適合性評価基準等の推奨基準に関して検討・整理されている。各推奨基準の概要は以下に示すとおりである。

1. ラベリングのための技術的な推奨基準 (NIST 文書 Section 2) :

消費者向けソフトウェアに関してセキュリティを担保するための技術的な評価項目と評価基準を定義している。文書では、ラベリング制度のオーナーに対し、ラベルを付与する際は、以下の 2 つにカテゴリ分けされたすべての基準をラベリング申請者に求めることを推奨している。

(a) 基本情報：ラベル申請者の情報、ラベルの対象スコープ、ラベル発行日、セキュリティ更新の有無、セキュリティサポートの最小期間、セキュリティ更新方法等、

対象となるソフトウェアに関する情報が明示されているか

(b) 安全なソフトウェア開発：SSDF が実装されているか、脆弱性が発見された際に利用者に通知されるか、多要素認証を使用しているか、機密情報はハードコードされていないか、強力な暗号化方式を使用しているか、ユーザーデータが適切に識別され保護されているか

2. ラベリングの基準 (NIST 文書 Section 3) :

ソフトウェアに対して付与されるラベルの種類を定義している。具体的なラベルとして、製品が推奨基準を満たしたことを示す唯一のラベルである「バイナリラベル」や、追加情報と併せて消費者に提供される「階層化ラベルアプローチ」¹⁷という種類・アプローチが記載されている。NIST は、ソフトウェアに対するサイバーセキュリティラベリングにおいて、「バイナリラベル」を採用することを推奨している。

3. 適合性評価基準 (NIST 文書 Section 4) :

ソフトウェアベンダーがラベリングの対象となるソフトウェアを定義し、ラベルを付与、そして消費者に対して適合宣言を実施するための基準を定義している。適合宣言のための基準として、供給者適合宣言 (Supplier's Declaration of Conformity) の概念に基づき、ソフトウェアベンダーが宣言に含めるべき項目及びソフトウェアが満たすべき基準がまとめられている。なお、第三者適合性評価プログラム等の外部の評価結果を活用して宣言を補足する際の必要な情報についてもまとめられている。

なお、現状で NIST が特定のラベリング制度を確立しているわけではなく、ラベリング制度に必要な要件や基準を特定することが本文書の目的であることに留意する必要がある。

2.1.1.12. Software Supply Chain Security Guidance Under Executive Order (EO) 14028 Section 4e

2022 年 2 月 4 日、米国 NIST は、大統領令を受け連邦政府機関が調達するソフトウェア¹⁸のサプライチェーンセキュリティ強化のための指針を示したガイダンス「Software Supply Chain Security Guidance Under Executive Order (EO) 14028 Section 4e」を発表した。特に、ソフトウェア調達に関わる連邦政府職員を対象に、SSDF (SP 800-218) の手法を実装する開発者からセキュアなソフトウェアを調達するために必要な 4 つの最低限の勧告事項を明記している。具体的な 4 つの勧告事項は以下のとおりである。

¹⁷ 「階層化ラベルアプローチ」とは、複数の階層に分けて製品に関する情報を開示するアプローチ（例：第一層として製品のパッケージにおいて代表的な情報を記載、第二層として製品に関するウェブサイトにより詳細な情報を記載）であり、厳密には「ラベルの種類」ではないことが明記されている。

¹⁸ 本ガイダンスの対象となる「連邦政府機関が調達するソフトウェア」には、ファームウェア、OS、アプリケーション、アプリケーションサービス（クラウドベースのソフトウェア）、ソフトウェアに使用される OSS 及びソフトウェアを使用する製品が含まれる。オンプレミス・クラウドの両方のソフトウェアが含まれるが、連邦政府機関によって開発されたソフトウェアや連邦政府機関が直接的に入手した OSS は対象外となる。

- **SSDFで定義された用語に基づき、安全なソフトウェア開発要件に関する共通言語を整理する。**
 - これにより、ソフトウェア開発者が連邦政府機関へ納入するための適合性を証明する際に、同じ共通言語を用いてコミュニケーションを取ることができる。
- **ソフトウェアのライフサイクルを通じて実践される、安全なソフトウェア開発の手法に関する証明を要求する。**
 - 今日のソフトウェアは非常に動的であるため、一つのソフトウェアに関して証明を求めるより、プロセスや手順を含んだ組織の開発手法を確認する方が一般的に価値がある。
- 安全なソフトウェア開発の対応に関する証明の要求に際して、第二者又は第三者による認証が必要であると判断されない限り、**第一者認証によるSSDF適合の証明を認める。**
- **適合性を示すためのアーティファクトを要求する場合、上位レベルのアーティファクトを要求する。**
 - 上位レベルのアーティファクトとは、組織のソフトウェア開発プラクティスを要約した文書であり、セキュアなソフトウェア開発のための方法、手順及びプロセスが記載された文書を指す。
 - 他方、ソフトウェア開発の過程で生成される脅威モデル、ソースコード、ソースコードの脆弱性スキャンレポート等は下位レベルのアーティファクトと定義される。

出所) NIST 文書に基づき三菱総合研究所作成

図 2-7 連邦政府機関がセキュアなソフトウェアを調達するための4つの勧告事項

ガイダンスでは、4つの勧告事項が調達するすべてのソフトウェアに適用されるために、各政府機関の調達プロセスに含まれるべきとしている。各政府機関の調達プロセスにこれらの勧告事項が含まれた場合、連邦政府の調達対象となるソフトウェアは、SSDFの手法を実装することが必須となる。

2.1.1.13. OMB Statement on “Enhancing The Security Of Federally Procured Software”

2022年3月7日、米国OMBは、政府調達ソフトウェアのセキュリティに関する声明「Enhancing The Security Of Federally Procured Software」を発表した。2022年2月にNISTより公開されたSSDFのVersion 1 (2.1.1.10 参照) 及び連邦政府機関が調達するソフトウェアのサプライチェーンセキュリティ強化のための指針を示したガイダンス (2.1.1.12 参照) を受け、大統領令以降に政府機関で調達されたソフトウェアに対して、これらのガイダンスを遵守するよう求める措置を講じているとの内容が発表された。また、OMBはSSDFを実践するベンダーがその旨を証明することの重要性を理解しており、その具体的な運用方法について2022年3月23日に公開ワークショップ¹⁹を開催して意見を聴取する旨が発表された。本ワークショップの開催に先立ち、ワークショップで意見を募集する対象となる以下の論点が公開された²⁰。

1. 連邦機関が調達するソフトウェアに関するセキュアなソフトウェア開発証明文書を取得し、保持するための理想的なプロセスはどのようなものか。
2. 適合性を評価するためのシステム、ツール、手順の成功例の中で、SSDFへの適用を検討すべきものはあるか。他の確立されたプロセスのどのような特徴を模倣することが最も重要か。適合性を証明するための特定の標準的な形式を推奨すべきか。
3. SSDFの要素で、認証よりも効果的な代替方法（例：適合性評価など）があるか。

¹⁹ NIST, Workshop to Inform Implementation Guidance for Federal Procurement of Secure Software
<https://www.nist.gov/news-events/events/2022/03/workshop-inform-implementation-guidance-federal-procurement-secure>

²⁰ OMB, Implementation Questions
<https://www.nist.gov/system/files/documents/2022/03/07/EO%204k%20implementation%20questions.pdf>

4. 適切な SSDF の実施状況を確認するために第三者認証が適切である場合を決定するために、どのようなリスクベースの要因を考慮すべきか。
5. ソフトウェアベンダーは、認証の対象となる製品や製品の境界をどのように明示すべきか。
6. ガイダンスに準拠するために、ベンダーは事前にどのような情報を必要とするか。

2.1.2. NTIA の Software Component Transparency に関連する取組動向の調査

米国 NTIA では、2018 年 7 月より官民のステークホルダー会合である「Software Component Transparency」を立ち上げ、ソフトウェアの脆弱性情報の理解と処理、成長する IoT マーケットへの対処、安全なソフトウェア開発ライフサイクルの促進等について議論を行っている。本会合には NTIA や NIST 等の行政機関のほか、ソフトウェアベンダー、通信プロバイダー、医療・ヘルスケア事業者、金融事業者、自動車メーカー、IoT 機器メーカー、医療機器メーカー等の企業が参画し、5 つのワーキンググループ（WG）において、SBOM の利活用促進に向けた議論が進められている。

1. Framing WG:

SBOM の定義と仕様に関して改善することで、SBOM 利用のスケラビリティを可能にすること、そしてその利用を促進することに取り組んでいる。SBOM 構成要素の名前付け、共有方法、悪用可能性と脆弱性の評価方法、整合性の保証、SaaS 向け SBOM 等も検討している。

2. Awareness and Adoption WG:

SBOM の認知と導入、導入促進を目的として、ターゲットの設定やアウトリーチ戦略の構築等に取り組んでいる。また、アウトリーチ目標を記載した文書の作成、関連する取組との調整、及び SBOM の採用に関する明確なビジネスケースの策定を行う。

3. Formats and Tooling WG:

既存のツールを用いた SBOM のカタログ化とフォーマット間の変換機能の開発に取り組んでいる。具体的には、SBOM の生成・利用を自動化する方法、既存の標準に対応したツールのカタログ化、及びフォーマット間の変換といった課題に取り組んでいる。

4. Healthcare Proof of Concept WG:

ヘルスケア分野での SBOM 活用に関する PoC を実施・主導することに取り組んでいる。業界のステークホルダーや、IT・セキュリティ業界のパートナー、Awareness and Adoption WG と連携している。

5. Automotive Proof of Concept Working Group:

自動車分野での SBOM 活用促進に取り組む WG であり、2021 年に新たに設置された WG であると考えられる²¹が、これまで本 WG を発行主体とした取組等は公開されていない。

²¹ INL, Energy Sector SBOM PoC Project Charter <https://inl.gov/wp-content/uploads/2021/05/SBOM-Energy-POC-Charter-May-19.pdf>

それぞれの WG において議論及び検討された内容に関する文書がこれまで多く発表されている。本事業で調査した本会合に関連する取組動向の調査対象の一覧を表 2-8 に示す。以降では、各取組の概要、その取組が与える影響等について示す。

表 2-8 NTIA の Software Component Transparency に関連する取組動向の調査対象

#	取組名・文書名	発行日	発行主体／ 関連する WG
2-1	SBOM Tool Classification Taxonomy ²²	2021 年 3 月 30 日	NTIA / Formats and Tooling WG
2-2	SBOM at a Glance ²³	2021 年 4 月 27 日	NTIA / Awareness and Adoption WG
2-3	Energy Sector SBOM PoC Charter ²⁴	2021 年 4 月 28 日	INL
2-4	Vulnerability-Exploitability eXchange (VEX) – An Overview ²⁵	2021 年 9 月 27 日	NTIA / Framing WG
2-5	How-To Guide for SBOM Generation ²⁶	2021 年 10 月 14 日	NTIA / Healthcare Proof of Concept WG
2-6	Healthcare SBOM Proof of Concept – Phase II Summary ²⁷	2021 年 10 月 14 日	NTIA / Healthcare Proof of Concept WG
2-7	SBOM Myths vs. Facts ²⁸	2021 年 11 月 1 日	NTIA / Awareness and Adoption WG
2-8	Software Suppliers Playbook: SBOM Production and Provision ²⁹	2021 年 11 月 17 日	NTIA / Formats and Tooling WG
2-9	Software Consumers Playbook: SBOM Acquisition, Management, and Use ³⁰	2021 年 11 月 17 日	NTIA / Formats and Tooling WG

2.1.2.1. SBOM Tool Classification Taxonomy

2021 年 3 月 30 日、NTIA は SBOM ツールの分類を示した文書「SBOM Tool Classification

²² https://www.ntia.gov/files/ntia/publications/ntia_sbom_tooling_taxonomy-2021mar30.pdf

²³ https://www.ntia.gov/files/ntia/publications/sbom_at_a_glance_apr2021.pdf

(日本語翻訳版) https://www.ntia.gov/files/ntia/publications/sbom_at_a_glance_ja.pdf

²⁴ <https://inl.gov/wp-content/uploads/2021/07/SBOM-Energy-POC-Charter-Final.pdf>

²⁵ https://www.ntia.gov/files/ntia/publications/vex_one-page_summary.pdf

²⁶ https://www.ntia.gov/files/ntia/publications/howto_guide_for_sbom_generation_v1.pdf

²⁷ https://www.ntia.gov/files/ntia/publications/healthcare_sbom_proof_of_concept_-_phase_ii_summary.pdf

²⁸ https://www.ntia.gov/files/ntia/publications/sbom_myths_vs_facts_nov2021.pdf

²⁹ https://www.ntia.gov/files/ntia/publications/software_suppliers_sbom_production_and_provision_-_final.pdf

³⁰ https://www.ntia.gov/files/ntia/publications/software_consumers_sbom_acquisition_management_and_use_-_final.pdf

Taxonomy」を発表した。この文書は、Formats and Tooling WG により策定され、SBOM ツールの利用目的・利用タイプが整理されている。具体的には、表 2-9 に示すとおり、ツールの利用目的を SBOM の作成・利用・変換の 3 つに分類し、それぞれの目的におけるツールのタイプが整理されている。

表 2-9 NTIA による SBOM ツールの分類

ツールの利用目的	ツールのタイプ	概要
作成 (Produce)	ビルド (Build)	SBOM をソフトウェア・アーティファクトの一部として自動的に生成するツール。ビルド自身の情報も含まれる。
	分析 (Analyze)	ソースファイルやバイナリファイルを解析し、アーティファクトと関連するソースを検査することで、SBOM を生成するツール。
	編集 (Edit)	SBOM を手動で生成・編集する人を支援するツール。
利用 (Consume)	可視化 (View)	SBOM を人間が読める形（絵、図、表、テキストなど）で内容を理解できるようにするツール。意思決定及びビジネスプロセスをサポートするために使用される。
	比較 (Diff)	複数の SBOM を比較し、その違いを明確に把握するためのツール。（例：ソフトウェアの 2 つのバージョンを比較する。）
	インポート (Import)	SBOM を検出・取得し、処理や分析のためにシステムにインポートするツール。
変換 (Transform)	変換 (Translate)	SBOM を、あるファイルタイプから別のファイルタイプへ、同じ情報を保持したまま変更するツール。
	マージ (Merge)	SBOM やその他のデータソースを、分析や監査の目的で結合するツール。
	他ツールのサポート (Tool support)	その他のツールでの SBOM 使用をサポートするツール。API、オブジェクトモデル、ライブラリ、トランスポート、その他の参照ソースの活用が想定される。

出所) NTIA 文書に基づき三菱総合研究所作成

2.1.2.2. SBOM at a Glance

2021年4月27日、NTIAはSBOMの活用方法や参考文献、エコシステムに存在するソフトウェアコンポーネントの透明性確保においてSBOMが果たす重要な役割等について概説した文書「SBOM at a Glance」を公開した。JPCERT/CCにより日本語訳された文書も公開されている。本文書では、SBOMとは、「ソフトウェアコンポーネントやそれらの依存関係の情報も含めた、機械処理可能なインベントリ」であり、「ソフトウェアコンポーネントやその依存関係をすべて表現している場合もある」ということが明記されている。加えて、「どこの部分が欠けているかという情報が含まれている場合もある」ほか、「OSSだけではなくプロプライエタリ・ソフトウェアに活用することもでき、広く一般に公開するほか関係者だけに提示するという使用方法もある」ことが明記されている。

SBOMに含める情報として、「ソフトウェアコンポーネントを確実に識別するための基本アトリビュート情報（Baseline Component 情報）を含めるべき」であることが明記されている。なお、この基本アトリビュート情報には、開発者（Author Name）、サプライヤー名（Supplier Name）、コンポーネント名（Component Name）、バージョン情報（Version String）、コンポーネントのハッシュ値（Component Hash）、識別子（Unique Identifier）、依存関係（Relationship）を組み合わせる使用することが推奨されているが、この情報だけで対応できないユースケースも存在するため、必要に応じてこれ以外の情報を活用する必要性も明記されている。そして、SBOMの作成について、ソフトウェア開発プロセスにおいて、ソフトウェア自体とともにSBOMを作成することが最も効率的であるとしている。

2.1.2.3. Energy Sector SBOM PoC Charter

2021年4月28日、INLが主導するエネルギー分野でのSBOM PoC推進に向け、INLはPoCの憲章である「Energy Sector SBOM PoC Charter」を発表した。本憲章では、PoCの目的や目標、PoCにおける主な検討課題等が示されている。具体的な内容は表2-10に示すとおりである。なお、エネルギー分野のPoCは2021年で終了し、終了のタイミングで継続性が判断された。判断の結果、2022年以降もPoCを継続することが決定し、フェーズ2のPoCが2022年2月より開始している³¹。

表 2-10 INL によるエネルギー分野の SBOM PoC の概要

目的	● エネルギー分野での SBOM の活用について調査し、SBOM 活用を促進するとともに、ソフトウェアコンポーネントの透明性を高める。 ● NTIA における SBOM の各 WG 活動の結果を活用しつつ、オープンな場において SBOM 導入のための技術やプロセス開発を促進する。

³¹ INL, Software Bill of Materials <https://inl.gov/sbom-poc/>

目標	● エネルギー分野でのサプライヤー、アセットオーナー（事業主体）、サードパーティベンダー等を<u>巻き込んだ多様な利害関係者を集めて、SBOMに関する知識と経験を共有する。</u> ● エネルギー分野における<u>ベンダーとアセットオーナーの間で SBOM 情報の作成と交換を促進する。</u> ● エネルギー分野において <u>SBOM を活用するためのユースケースを調査する。</u> ● <u>SBOM 技術の実装上の課題と緩和策を特定する。</u>
主な実施内容 ³²	● テストケースの作成やデモンストレーション等を含みながら、<u>SBOM の作成、配布、使用について議論するワークショップを開催する。</u> ● エネルギー業界へソフトウェア等を提供するサプライヤーによって作成された SBOM、及びサプライチェーンリスクの軽減のために<u>アセットオーナーに対して作成された SBOM を使用する。</u> ● PoC 活動とそこから得られた知識や課題をまとめたレポートを 1 つ以上作成する。
PoC における主な検討課題	● PoC の関係者は多様であり、SBOM に関する知識レベルが様々であるため、関係者に一律の行動方針を定めることは難しいこと。 ● 提供された情報が悪用される可能性があることに不安を感じているサプライヤーが存在すること。 ● 容易に SBOM を作成・利用できるようになるためには、具体的なユースケースや技術的な障壁を見極める必要があること。 ● SBOM 作成のツールは既に開発されているが、脆弱性管理やライセンス管理に係る SBOM ツールは限定的であること。 ● SBOM の作成、配布、利用のために実行可能な手順に到達するには、多様な視点と関心を調整する必要があること。
備考	● 今回の PoC は NTIA によって開発された SBOM フォーマットや概念に基づいて実施する。 ● 商用ツールが SBOM の活用に不可欠であるとしつつ、今回の PoC では特定の商用ツールの推奨や宣伝は実施しない

出所) INL 文書に基づき三菱総合研究所作成

2.1.2.4. Vulnerability-Exploitability eXchange (VEX) – An Overview

2021 年 9 月 27 日、NTIA は、特定のソフトウェアコンポーネントが脆弱性の影響を受けるかどうかを判断できる指標である VEX (Vulnerability Exploitability eXchange) について、その概要を説明する文書「Vulnerability-Exploitability eXchange (VEX) – An Overview」を公開した。文書では、VEX は、特定の製品に存在する脆弱性の状態を表すもので、文書では以下の 4 段階で状態を表す方針が示されている。

³² 新しいツールや技術の開発、特定商用ツールの使用はスコープ外とされている。

- **影響を受けない (Not affected) :**
脆弱性について対策の必要はない。
- **影響を受ける (Affected) :**
脆弱性を修正する又は対処することが推奨される。
- **修正済み (Fixed) :**
対象製品のバージョンには、脆弱性に対する修正が施されている。
- **調査中 (Under investigation) :**
対象製品のバージョンが脆弱性の影響を受けるかどうかは、まだ判明していない。

VEX の主な用途は、ユーザー（オペレータ、開発者、サービスプロバイダーなど）に対し、コンポーネントが特定の脆弱性の影響を受けるかどうか、そして影響を受ける場合には是正するために推奨されるアクションがあるかどうかについての追加情報を提供することであるとしている。

2.1.2.5. How-To Guide for SBOM Generation

2021 年 10 月 14 日、NTIA より SBOM 生成の手引きである「How-To Guide for SBOM Generation」が公開された。本手引きは、NTIA によるヘルスケア分野の SBOM PoC を通じて策定されたが、ヘルスケア分野だけでなくあらゆる業界における SBOM 生成においての活用が期待されている。手引きでは、SBOM 生成のための情報収集方法と、具体的な SBOM 生成方法の 2 つの観点から、具体的な手引きが記載されている。

2.1.2.5.1. SBOM 生成のための情報収集方法

手引きでは、SBOM 生成プロセスを未確立の組織や確立中の組織に向けて、SBOM 生成のために必要な情報や、当該情報を収集する方法が示されている。具体的には、以下の 4 項目の重要性が明記されている。

- **組織内の製品チームやエンジニアリングチームのリソースを活用する :**
製品チームやエンジニアチームのリソース及びドキュメントを活用することで、SBOM 情報を収集することができる。
- **ソフトウェア構成分析 (SCA) ツールやバイナリコード分析ツールを使用する :**
ソフトウェアコンポーネントを特定するツールを使用することを検討する。ただし、ツールにはメリット・デメリットがあることに留意する必要がある。
- **法律・ライセンス・コンプライアンスに関する情報を活用する :**
多くの組織では、ライセンス要件を遵守するため、ライセンスデータベースや紙のファイルを管理している。また、構成管理データベース (CMDB) やアセットマネージャーなどのツールを使用し、ソフトウェアやそのライセンスの情報をデータベース化し、管理している。管理しているデータベースを活用することで、SBOM 情報を収集できる可能性がある。
- **SBOM に必要な基本情報を収集する :**
SBOM においては、SBOM の生成者、コンポーネントのサプライヤー名、バージョン、コンポーネント間の関係などの情報が基本となる。

組織内の製品チームやエンジニアリングチームのリソースを活用することについて、手引きでは、製品チームやエンジニアチームのリソース及びドキュメントを活用することで、SBOM 情報を収集することができるとしている。具体的には、エンジニアリングマネージャー、製品要件のマネージャー又は製品マネージャーのほか、ソフトウェア構成に関する詳細な情報が明記される場合が一般的である製品開発ドキュメントを活用することが推奨されている。

ソフトウェア構成分析 (SCA) ツールやバイナリコード分析ツールを使用することについて、エンジニアリングチームや製品管理チームの支援が得られない場合に、ソフトウェアをスキャンして含まれるコンポーネントを特定するツール (SCA ツールやバイナリコード分析ツール) の使用を検討すべきとしている。ただし、ツールにはメリットとデメリットがあるとしており、メリットとしては、標準形式を用いて SBOM を自動生成することが可能であるほか、様々な脆弱性情報データベースを参照し、既知の脆弱性とコンポーネントを関連付けることが可能であることを挙げている。他方、デメリットとしては、含まれているすべてのコンポーネントを正しく識別できない可能性があることや、コンポーネントの誤検出を引き起こす可能性があることが示されている。

法律・ライセンス・コンプライアンスに関する情報を活用することについて、多くの組織では、ライセンス要件を遵守するため、ライセンスデータベースや紙のファイルを管理している。ライセンスは法的、財務的、知的財産に関する大きなリスクの原因となるため、多くの企業がコンプライアンスの徹底を図っているが、ソフトウェアに含まれるすべてのコンポーネントのライセンスは、サプライヤーが、そのソフトウェアを使用する組織に提供する必要がある。多くの企業は構成管理データベース (CMDB) や資産マネージャーなどのツールを使用し、ソフトウェアやそのライセンスの情報をデータベース化し管理している。手引きでは、データベースが適切に管理されている場合、SBOM データの適切なソースとなる可能性があるとしている。一方で、ライセンス情報に誤りがある可能性も存在するため、SBOM 情報の検証用として使用することも一案であるとしている。

SBOM に必要な基本情報を収集することについて、「SBOM at a Glance」で示された基本アトリビュート情報である、開発者 (Author Name)、サプライヤー名 (Supplier Name)、コンポーネント名 (Component Name)、バージョン情報 (Version String)、コンポーネントのハッシュ値 (Component Hash)、識別子 (Unique Identifier)、依存関係 (Relationship) を収集すべきとしている。

2.1.2.5.2. 具体的な SBOM 生成方法

手引きでは、SBOM 生成において必要な情報や具体的な生成方法も示されており、具体的には、以下の 6 項目の重要性が示されている。

- **ソフトウェア ID の基本要素を特定する：**
コンポーネントを特定するために必要な基本要素は、サプライヤー名、コンポーネント名、バージョンである。
- **ソフトウェア ID を一意に決定する：**
パッケージ URL (purl) 構文を使用してソフトウェア ID を生成することで、ソフトウェア ID の一意性を保証することができる。

- **SBOM の完全性を確保する：**
上流工程のコンポーネントにおける SBOM が存在しない場合、上流工程のコンポーネントの把握状況を踏まえ、SBOM を生成する必要がある。
- **SBOM ドキュメント ID を特定する：**
ソフトウェア ID だけでなく、SBOM ドキュメントの ID も一意に特定する必要がある。
- **プライマリーコンポーネントでない場合の SBOM 生成方法を検討する：**
対象がプライマリーコンポーネントでない場合は、最終製品の SBOM に直接情報を記述するか、上流工程で作成された外部の SBOM へ参照するかの方法がある。
- **医療機器システムの SBOM を生成する場合、医療機器をプライマリーコンポーネントとして扱う：**
医療機器はプライマリーコンポーネントであり、そのサブシステムをプライマリーコンポーネントでないものと位置づけられる。SBOM 生成者は、サブシステムを医療機器の SBOM に直接情報を記述するか、外部参照するかを決定する必要がある。

手引きでは、SBOM 生成に関する課題の 1 つは、SBOM 全体で、同じコンポーネントに対して一貫性のある一意のソフトウェア ID を生成することであるとしている。これに対して、一貫性と一意性を満たすソフトウェア ID を取得する必要がある、SBOM 生成者はサプライヤー名、コンポーネント名、バージョン情報の 3 つの情報をプライマリーコンポーネントに定義する必要があるとしている。対象がプライマリーコンポーネントでない場合、コンポーネント用に公開された SBOM 情報から取得する方法のほか、サプライヤーから取得する方法、既存のシステム（NVD 等）から取得する方法が想定されるとしている。

ソフトウェア ID を一意に決定することについて、手引きでは、パッケージ URL（purl）を使用することで、サプライヤーは一意の ID として提供することができるとしている。パッケージ URL を使用した ID は、ソフトウェア ID の基本要素から構成されるため、基本要素が一意である場合、パッケージ URL も一意となり、ソフトウェア ID として使用できるとしている。

SBOM の完全性を確保することについて、手引きでは、上流工程のコンポーネントの SBOM が存在しない場合、下流又は最終製品メーカー（OEM）は、不完全な情報で SBOM を生成する必要があるため、ヘルスケア分野での PoC では、上流工程のコンポーネントに関する把握状況を 4 段階に分類し、分類ごとに SBOM 生成に取り組んだとしている。

<プライマリーコンポーネントの場合>

- **Unknown（未知）：**
含まれるコンポーネントを特定する試みが実行されなかった場合。
- **Partial（部分的）：**
含まれるコンポーネントを特定するため、SCA が実行されたが、コンポーネントのリストが不完全である可能性がある場合。
- **Known（既知）：**
含まれるすべてのコンポーネントが特定され、SBOM に組み込まれている場合。
- **Root（ルート）：**
プライマリーコンポーネントに含まれるコンポーネントがない場合。

<プライマリーコンポーネントではない場合>

- **Unknown (未知) :**
含まれるコンポーネントを特定されず、SCA も実行されなかった場合。
- **Partial (部分的) :**
SBOM が取得され、SBOM によって記述されたコンポーネントが「部分的」と判断されるか、コンポーネントを特定するために SCA が実行された場合。
- **Known (既知) :**
SBOM が取得され、SBOM によって記述されたコンポーネントは「既知」と判断された場合。
- **Root (ルート) :**
SBOM が取得され、SBOM によって記述されたコンポーネントは「ルート」と判断された場合。

SBOM ドキュメント ID を特定することについて、手引きでは、同じバージョンのソフトウェアに対して複数のバージョンの SBOM ドキュメント (SBOM 出力結果) が存在する可能性があるため、SBOM ドキュメントを識別することが重要であるとしている。SBOM ドキュメントを識別することで、例えば、新しいバージョンの SBOM を生成した場合に、古いバージョンの SBOM ドキュメントの誤った情報を修正できるようになる。手引きでは、SBOM ドキュメント ID に必要な要素として、SBOM 生成者、SBOM ドキュメント名、SBOM ドキュメントバージョンを含めるべきとしており、これらに加えて SBOM ドキュメントが生成された時間を取得することは SBOM のバージョン管理に適しており、ソフトウェアコンポーネントの最新かつ信頼できる SBOM を判断するのに役立つとしている。なお、SBOM ドキュメント ID のルールについて、手引きでは、ソフトウェア ID と同様に SBOM ドキュメント名と SBOM ドキュメントバージョンを設定するために論理的な規則を使用する必要があるとしている。

プライマリーコンポーネントでない場合の SBOM 生成方法を検討することについて、手引きでは、プライマリーコンポーネントでない場合の SBOM 情報は、最終製品の SBOM に直接記述されるか、上流製品の SBOM へ参照されて提供されることが望ましいとしている。

最後に、医療機器システムの SBOM を生成する場合に、医療機器をプライマリーコンポーネントとして扱うことについて、ヘルスケア分野の PoC を通じて、医療機器はプライマリーコンポーネントであり、サブシステムはプライマリーコンポーネントではないと位置づけられることを確認したとしている。そのため、医療機器に関わる SBOM 生成者は、サブシステムを医療機器の SBOM に直接含めるか、外部参照させるか決定する必要があるとしている。

2.1.2.6. Healthcare SBOM Proof of Concept – Phase II Summary

2021 年 10 月 14 日、NTIA はヘルスケア分野での SBOM PoC のフェーズ 2 の結果をまとめた文書「Healthcare SBOM Proof of Concept – Phase II Summary」を公開した。ヘルスケア分野の SBOM PoC のフェーズ 1 は 2018 年 7 月から実施され、今回結果が取りまとめられたフェーズ 2 は 2019 年 10 月から 2021 年 10 月まで実施された。フェーズ 2 の PoC は、主に 3

つの反復（Iteration）を繰り返すことで実施された。各反復の主な実施事項や成果は以下のとおりである。

< Iteration 1>

- 医療機器メーカー³³、医療機関等³⁴の参加者を拡大し、SBOM のユースケースを作成
- SBOM の SPDX 互換性の調査
- SBOM に必要な基本情報の議論

< Iteration 2>

- SBOM ドキュメントバージョンに関する議論
- SBOM コンポーネントの完全性に関する議論
- パッケージ URL（purl）を使用したソフトウェア ID の構築方法の検討
- ソフトウェア ID の議論として、共通コンポーネントのリスト化や SBOM 全体での共通コンポーネントの調整を実施
- SBOM 情報の記述方法の議論として、プライマリーコンポーネントで無い場合に、SBOM へ直接記述するか、外部の SBOM へ参照するかの議論
- メインシステムとサブシステムでの SBOM 生成の位置づけの議論

< Iteration 3>

- CSAF（Common Security Advisory Framework）形式を使用した VEX ユースケースの調査
- 複数の一意の識別子の議論を実施し、purl 及び CPE（又は CPE 形式）の使用を検討
- 対応するデバイスを識別するためのファイル命名規則の確立
- SBOM 生成のための手引きの策定

今後、フェーズ 3 として以下の課題に取り組むことが言及されている。

- ヘルスケア分野での SBOM 利活用の促進
- 小規模の医療機関など、新たな属性の参加者を拡大
- SBOM 共有の自動化
- VEX のさらなる議論
- サービス終了・サポート終了の情報共有に関する議論
- フェーズ 2 での未実施事項の検討

2.1.2.7. SBOM Myths vs. Facts

2021 年 11 月 1 日、NTIA は SBOM のメリットを正しく示すことを目的に、SBOM に関する代表的な誤解（神話）と、その誤解を解くための事実を整理した文書「SBOM Myths vs.

³³ Abbott、Becton Dickenson、Medtronic、Philips、Roche、Siemens Healthineers、Thermo Fisher Scientific

³⁴ Cedars-Sinai, Christiana Care, Cleveland Clinic, Intermountain Healthcare, Mayo Clinic, New York Presbyterian, Sutter Health, Univ. of Virginia

Facts」を公開した。本文書で記載された SBOM に関する誤解と事実は以下に示すとおりである。

表 2-11 SBOM に関する誤解と事実

誤解	事実
SBOM は、攻撃者にとよる攻撃を支援する	<u>SBOM は攻撃に利用可能であるが、SBOM により透明性を確保することによる「攻撃者からの防御」におけるメリットの方が大きい。</u>攻撃者にとっての SBOM やソフトウェアの透明性は、次のように位置づけられる。 ● 一般的に攻撃者は SBOM を必要としない。例えば、WannaCry によるランサムウェア攻撃は、SBOM 情報が攻撃のための前提条件ではなかった。 ● 攻撃のためのツールは、SBOM を活用せずとも、攻撃対象となるソフトウェアコンポーネントを簡単に特定できる。
SBOM だけでは、有用・実用的な情報を得ることができない	<u>SBOM は、ソフトウェアの開発者、購入者、運用者をサポートする。</u>例えば、攻撃を受けている際、SBOM を使用することで、攻撃の影響を受けているか、攻撃の影響範囲はどこかを容易に判断できる。コンポーネント情報があることでソフトウェアの透明性が向上し、管理が容易となり、開発、購入、運用段階以外での活用も可能となる。
SBOM は公開しなければならない	<u>SBOM を公開する必要はない。</u> <u>「SBOM の作成」と「SBOM を前向きに活用できる人との共有」は別の観点であり、SBOM 作成者の判断で SBOM を共有することができる。</u> 業界における規制や法定要件により、SBOM へのアクセスを求められる場合がある点には留意する必要がある。また、大統領令でも SBOM の公開は SBOM 作成者の判断であり、必須ではないことが明確に記載されている。「ソフトウェア購入者に SBOM を直接提供するか」、「公開 Web サイトに SBOM を公開するか」という点については、SBOM 作成者が判断できる。

誤解	事実
SBOM は知的財産や企業秘密を露呈してしまう	<u>SBOM はソフトウェアに含まれているコンポーネントの概要であり、特許やアルゴリズムは含まれておらず、知的財産を公開するものではない。</u> SBOM の内容や知的財産に関して考慮すべき点は以下のとおり。 ● SBOM は単なる「材料の一覧」であり、特許やアルゴリズムのような「レシピ」とは異なる。 ● 第三者が開発したコンポーネントの特許やアルゴリズムなどの知的財産は、コンポーネントの開発者又は著作権所有者に帰属する。 ● 使用するソフトウェアコンポーネントのライセンス条項において、情報開示を要求されることが近年増加している。 ● SBOM には、ソフトウェアのソースコード自体は含まれない。 ● 契約、法的等の要件で、特定のコンポーネント開示が禁止されている場合がある。この際、「既知の不明なコンポーネント」の存在を示す必要がある。
SBOM を活用するためのサポートは存在しない	<u>ソフトウェア構成分析 (SCA) ツールは、一部の分野では、10 年以上にわたって企業内で使用されてきた実績がある。</u>ソフトウェアの透明性に関しては、NTIA の活動、大統領令、SBOM フォーマットの標準化などの活動が進んでいるほか、一部の分野では、ソフトウェアの透明性について 5 年以上にわたって議論や PoC の取組が進められており、他分野での導入をサポートしている。

出所) NTIA 文書に基づき三菱総合研究所作成

2.1.2.8. Software Suppliers Playbook: SBOM Production and Provision

2021 年 11 月 17 日、NTIA はソフトウェアサプライヤーを対象とした SBOM 生成に関するプレイブック「Software Suppliers Playbook: SBOM Production and Provision」を公開した。本プレイブックでは、「SBOM 作成手順」、「SBOM 作成にあたって考慮すべき事項」及び「SBOM に関する補足事項」の 3 つの事項についてまとめられている。各事項の概要は以降に示すとおりである。

2.1.2.8.1. SBOM 作成手順

ソフトウェア開発組織は多様であり、様々なソフトウェアやシステムに対して SBOM を作成することが必要である。開発組織は様々なツールやプロセスを用いて、SBOM を作成することが可能である。SBOM 作成手順は一般的に以下の手順となる。

1. コンポーネントの特定

対象となるソフトウェアに含まれるソフトウェアコンポーネントを特定する。

2. コンポーネント情報を取得

特定したソフトウェアコンポーネントに関する情報を取得する。

3. SBOM 形式への出力

コンポーネント情報を、構造化された SBOM 形式へ出力する。

4. SBOM の検証

作成した SBOM フォーマットが有効であるかを検証し、コンポーネントに最低限の属性情報が存在することを確認する。

2.1.2.8.2. SBOM 作成にあたって考慮すべき事項

- **SBOM 作成の自動化**

ビルド前のソースレベルの SBOM の生成にあたっては、ソフトウェアバージョン管理ツールや CI/CD パイプラインなどを活用することで、SBOM を自動作成することが可能となる。

- **コンテナイメージに対する SBOM の作成**

コンテナイメージには、様々なソフトウェアアプリケーションや、様々なレイヤーに組み込まれたアーティファクトが含まれる。そのため、全レイヤーの全ソフトウェアを特定し、SBOM に記述する必要がある。

- **SBOM 作成日時の明確化**

ビルド後に作成された SBOM の場合、いつ SBOM が作成されたかを明確化するために、SBOM の作成日時に関する情報を含める必要がある。

- **SBOM に含まれる情報の明確化**

アプリケーションとともに利用者に提供される追加のコンポーネント情報（ダイナミックリンクライブラリ、共有ライブラリ 等）が SBOM に含まれるか、利用者に明示する必要がある。

- **外部サービスの明確化**

アプリケーションが機能を実行するために、インターネットサービスを呼び出す場合、当該サービスに関する情報を可視化する必要がある。ただし、これは検討段階であるため、SBOM の最小要素としては含まれていない。

2.1.2.8.3. SBOM に関する補足事項

- **SBOM の知的財産／機密性**

SBOM 情報は中間サプライヤーを介して最終利用者に提供される必要がある。SBOM の配布を妨げるのではなく、契約上の機密情報として SBOM を扱うように機密保持体制を構築することが望まれる。

- **SBOM フォーマットの検証**

SBOM のフォーマットが有効であるか（必要な情報が存在し、構造化されているか）を確認する。活用できるツールの例は以下のとおり。

- ・ SPDX Online Tool³⁵ : SPDX 形式の検証ツール
- ・ SWID Tools³⁶ : SWID 形式の検証ツール
- ・ CycloneDX CLI Tool, Web Tool³⁷ : CycloneDX 形式の SBOM 検証ツール

- **コンポーネント情報の検証**

SBOM に含まれるコンポーネント情報の確からしさを検証する。活用できるフレーム

³⁵ <https://tools.spdx.org/app/>

³⁶ <https://pages.nist.gov/swid-tools/>

³⁷ <https://github.com/CycloneDX/cyclonedx-cli>, <https://cyclonedx.org/tool-center/>

ワークの例は以下のとおり。

- ・OWASP SCVS：ソフトウェアコンポーネントの評価や改善方法の参考となるフレームワーク
- ・OpenChain (ISO/IEC 5230:2020)：ソフトウェアコンポーネントの正確な特定と監視に必要なとなるプロセス管理標準

2.1.2.9. Software Consumers Playbook: SBOM Acquisition, Management, and Use

2021年11月17日、NTIAは前項のソフトウェアサプライヤー向けのプレイブックと合わせ、ソフトウェア利用者を対象としたSBOM利用に関するプレイブックである「Software Consumers Playbook: SBOM Acquisition, Management, and Use」公開した。本プレイブックでは、「サプライヤーからSBOMを取得する際の注意点」、「SBOM活用のプロセス及びプラットフォーム」、「SBOMの知的財産及び機密保持」に関する注意点等がまとめられている。各事項の概要は以降に示すとおりである。

2.1.2.9.1. サプライヤーからSBOMを取得する際の注意点

- **SBOM取得のタイミング**

ソフトウェア利用者は、以下のような場合に、SBOMを取得することができる。

- ソフトウェアやサービスの契約や調達時
- プロプライエタリ・ソフトウェアのダウンロード時
- ソフトウェアの開発・提供に係る専門サービスの契約や調達時
- 開発時などの社内展開用として、OSSアプリケーション又はコンポーネントの取得時
- デバイスのネットワーク接続時（SBOM検出プロセスが自動で実行する場合）

- **SBOMの対象となるソフトウェアの範囲**

ソフトウェアの定義は以下のように様々である。

- 単一のアプリケーション
- 外部と依存関係のあるアプリケーション
- ソフトウェアコンテナ
- 複数のエンドポイントを持つシステム

利用者は、SBOMの対象となるソフトウェアを確認する必要がある。

- **ソフトウェアコンポーネントの特定**

コンポーネントを正確に特定することで、脆弱性とのマッピングや、構成管理・ソフトウェア資産管理の曖昧さの排除が可能となる。

2.1.2.9.2. SBOM活用のプロセス及びプラットフォーム

- **SBOMの活用可能性**

SBOMの活用が組織内で十分に成熟すると、以下のプロセスやプラットフォームにおいて、SBOMを効果的に活用できる。

- 構成管理データベース (CMDB)
- ソフトウェア資産管理 (SAM)

- セキュリティオペレーションセンター（SOC）
- 調達に関するワークフロー（調達前調査、サプライヤー管理、サードパーティ/コンプライアンスのリスク管理など）
- ソフトウェア・サプライチェーンリスク評価管理

- **SBOM の継続的な監視**

利用者は、脆弱性対処ステータスについてサプライヤーと情報共有を継続することで、脆弱性の状況認識に係る利用者とサプライヤーとのギャップを取り除くとともに、SBOM の継続的な信頼性確保に寄与する。

- 脆弱性を検出する前の情報共有
- 脆弱性を検出し、脆弱性に対処する前の情報共有
- 脆弱性対処後、消費者へ共有する前の情報共有

2.1.2.9.3. SBOM の知的財産及び機密保持

ベンダー、請負業者、OSS コミュニティ等、どの立場によって SBOM が提供されるかで、SBOM の知的財産や機密保持に対する考え方は異なる。

- **ベンダーや請負業者によって SBOM が提供される場合**

SBOM 自体に適用される守秘義務条項は契約上、明示的に定義される。ソフトウェア利用者の視点では、SBOM を内部目的で使用する事が許可されるべきである。

- **OSS コミュニティによって SBOM が提供される場合**

OSS コミュニティが提供する SBOM は、ライセンスの下で明確に位置づけられるべきである。利用者は、SBOM に関する OSS のライセンスを確認する必要がある。

- **中間サプライヤーへ SBOM を提供する場合**

SBOM 利用者が中間サプライヤーである場合、最終消費者へ提供する SBOM が、中間サプライヤーへ提供される SBOM の機密保持条件を満たしていることを確認する必要がある。SBOM の知的財産や機密性の規定により中間サプライヤーのコンポーネントの特定が妨げられる場合、最終消費者のサプライチェーン透明性が損なわれる可能性がある。特に、省略、改訂及び「既知の未知」を特定するために、これらの規定の存在を SBOM で伝える必要がある。

2.1.3. その他の米国政府機関における取組動向の調査

これまで、米国大統領令を踏まえた連邦政府機関の取組や、NTIA における Software Component Transparency に関する取組について概説してきた。以降では、その他の米国政府機関における取組動向として、CISA における取組、FDA における取組及び DoD における取組を説明する。

2.1.3.1. CISA における取組

CISA は国家的リスクに関する助言を行う組織であり、将来に向けてより安全で強靱なインフラ設備を構築するために、ソフトウェアを安全に利活用するための文書の発表などを行っている。CISA の取組のうち、本報告書では以下の取組について概説する。

1. Defending Against Software Supply Chain Attacks³⁸
2. Bad Practices³⁹
3. Binding Operational Directive 22-01: Reducing the Significant Risk of Known Exploited Vulnerabilities⁴⁰
4. SBOM-a-rama⁴¹

2.1.3.1.1. Defending Against Software Supply Chain Attacks

2021 年 4 月 26 日、米国 CISA はソフトウェア・サプライチェーンにおける推奨事項を示した文書「Defending Against Software Supply Chain Attacks」を公開した。本文書では、ソフトウェア・サプライチェーンリスクの概要や事例を示すとともに、サプライチェーンリスクの特定・評価・軽減に向け、NIST の C-SCRM（サイバーサプライチェーンリスク管理）プログラムや SSDF（セキュアソフトウェア開発フレームワーク）に基づき、ソフトウェアの利用者及びベンダーが実施すべき推奨事項を示している。

ソフトウェア利用者が実施すべき推奨事項として、ソフトウェアを調達し利用する組織は、他の ICT 製品・サービスと同様に、リスク管理プログラムに基づいたソフトウェアの利用を検討すべきとしている。リスク管理プログラムでは、C-SCRM プログラムのアプローチを採用すべきであり、これにより、ソフトウェア・サプライチェーンリスクの緩和・対応を効率化することができるとしている。具体的には、NIST が提案する以下の 8 つのプラクティスに基づき、ソフトウェアに対する C-SCRM アプローチを確立すべきとしている。

1. 組織全体で C-SCRM を統合する。
2. 正式な C-SCRM プログラムを確立する。
3. 重要コンポーネントとサプライヤーを把握し、管理する。
4. 組織のサプライチェーンを把握する。
5. 主要なサプライヤーと緊密に連携する。
6. 主要なサプライヤーをレジリエンス強化及び改善の活動に巻き込む。
7. サプライヤーとの関係を通じて、サプライヤーの評価・監視を行う。
8. ライフサイクル全体の計画を構築する。

また、脆弱なソフトウェアコンポーネントが入り込んでしまった際にそれを緩和するための脆弱性管理プログラムの採用、構成管理、ファイアウォールや不正侵入検知/防御システムによる通信管理、組織の危機管理計画におけるソフトウェアの考慮等を実施すべきであるとしている。

ソフトウェアベンダーが実施すべき推奨事項として、通常業務においてソフトウェア開発ライフサイクル（SDLC）を実践することが推奨されている。また、ベンダーが自社のソフトウェアに関するリスクを緩和するために、SDLC に対して安全なソフトウェア開発手法を統合する必要がある、SSDF を SDLC に統合することで、悪意あるコンポーネントや脆弱性がソフトウェア・サプライチェーンに入り込むことを防ぐことができるとしている。そして、ソフトウェアベンダーは、安全なソフトウェア開発を行うために、以下の準備を行うことが必要であると位置づけている。

³⁸ https://www.cisa.gov/sites/default/files/publications/defending_against_software_supply_chain_attacks_508_1.pdf

³⁹ <https://www.cisa.gov/BadPractices>

⁴⁰ <https://cyber.dhs.gov/bod/22-01/>

⁴¹ <https://www.cisa.gov/cisa-sbom-rama>

- ソフトウェア開発のセキュリティ要件を定義する。
- SDLC における SSDF の役割と責任を確立する。
- 開発やセキュリティに関するツールチェーンを自動化する。
- ソフトウェアのセキュリティ基準と、セキュリティチェックに必要なデータを収集するためのプロセスを確立する。

また、NIST の推奨事項等を参考に、開発時にセキュリティに関して緩和策を検討するほか、パッチ適用が可能なソフトウェアの開発、SBOM 等のソフトウェア部品のインベントリの作成・提供を行うべきであるとしている。加えて、発見された脆弱性に対して可能な限り迅速に緩和策を提供することや、検出された脆弱性を分析し、その根本原因を特定すること、SDLC 全体の改善を図ることを求めている。

2.1.3.1.2. Bad Practices

2021 年 6 月 24 日、米国 CISA は、重要インフラや国家の重要機能におけるサイバーセキュリティ対策において、避けるべき 3 つのバッドプラクティスを示した文書である「**Bad Practices**」を公表した。3 つのバッドプラクティスを実施している場合、国家安全保障や経済的な影響、国民の生命や安全に対して影響を及ぼす可能性があることが示唆されている。

ソフトウェア管理に関するバッドプラクティスとして、重要インフラや国家の基幹機能のサービスにおいてサポートされていない（又は製造中止の）ソフトウェアを使用することが挙げられており、インターネットからアクセス可能な場合はさらに危険度が増すことが明記されている。なお、現状は 3 つのバッドプラクティスが記載されているが、今後追加される可能性があることが示されている。

2.1.3.1.3. Binding Operational Directive 22-01: Reducing the Significant Risk of Known Exploited Vulnerabilities

2021 年 11 月 3 日、米国 CISA は悪用された既知の脆弱性に対するリスクを低減するための「拘束力のある指令」である「**Binding Operational Directive 22-01: Reducing the Significant Risk of Known Exploited Vulnerabilities**」を発表した。連邦政府機関における脆弱性管理手順のレビューと更新を 60 日以内に行うことを指示したほか、「既知の悪用された脆弱性カタログ（**Known Exploited Vulnerabilities Catalog**）」⁴²を公開し、これまでに悪用が確認されている重大な脆弱性をカタログとしてリスト化したうえで、カタログ上で記載された期日までに連邦政府機関が各脆弱性を修正することを要求した。カタログには、Windows プリントスプーラーの脆弱性や Kaseya VSA の脆弱性等、既に悪用が確認された脆弱性の中でも、米国連邦政府に重大なリスクをもたらすと評価された脆弱性が含まれており、日々更新されている。

2.1.3.1.4. SBOM-a-rama

2021 年 12 月 15 日から 12 月 16 日にかけて、米国 CISA は SBOM に関するオンラインイベント「**SBOM-a-rama**」を開催した。本イベントは SBOM コミュニティの醸成を目的としており、1 日目は初級者向けに SBOM の情報を提供することを目的として、SBOM に関する

⁴² <https://www.cisa.gov/known-exploited-vulnerabilities-catalog>

複数の観点について有識者から講演及び有識者に対する Q&A が実施された。本イベントの 1 日目の講演アジェンダや講演者は以下に示すとおりである。

1. **SBOM に関するモチベーション**
SBOM の歴史、SBOM の役割や効果、SBOM 利用にあたっての実務者の視点、SBOM の定義について、計 5 件の講演。
※ 登壇者：CISA, Schneider Electric, New York-Presbyterian, CERT/CC 等
2. **SBOM のフォーマット及びツール**
SBOM ツールの分類法、SPDX の概要、CycloneDX の概要について、計 3 件の講演。
※ 登壇者：Linux Foundation, Microsoft, OWASP
3. **SBOM の実装**
NTIA が公開したソフトウェアサプライヤー及びソフトウェア利用者のためのプレイブックの概要、VEX (Vulnerability-Exploitability Exchange) の概要、大統領令で定義された SBOM の「最小要素」の概要について、計 3 件の講演。
※ 登壇者：Ion Channel, 独 BSI, CISA
4. **SBOM に関する PoC**
ヘルスケア分野、エネルギー分野及び自動車分野の SBOM PoC の概要について、計 3 件の講演。
※ 登壇者：Siemens, INL, Hitachi America

2 日目は、日本の SBOM に関する取組状況について経済産業省より発表した後、SBOM の利活用に向けた課題について参加者による議論が実施され、様々な観点に基づく課題が抽出・整理された。抽出・整理された主要な課題は表 2-12 に示すとおりである。イベントのまとめにおいて、今後整理した課題を優先度付けし、解決に向けた検討を推進する旨が発表された。

表 2-12 SBOM-a-rama の 2 日目の議論により抽出された主要な課題（抜粋）

観点	議論により抽出された主要な課題（抜粋）
クラウド	● クラウドサービスに対する SBOM ユースケースの作成 ● クラウドネイティブコンポーネントに対する SBOM 生成方法
データマネジメント	● SBOM 生成に必要なデータの提供方法 ● OSS コミュニティも含めた適切なデータの統合方法
ツール	● コンテナ環境、仮想環境、AI・機械学習等に対する SBOM 生成ツール ● SBOM 共有・活用ツール
普及・促進	● SBOM 活用に向けた教育 ● 各国の規制基準への適用 ● SBOM の出所と明確化や信頼性の確保

観点	議論により抽出された主要な課題（抜粋）
共有・交換	● SBOM のアーカイブを残す方法 ● SBOM 共有にあたっての信頼性や完全性の確保
技術導入	● SBOM のハッシュ化方法や署名方法
他分野との融合	● ファームウェアやハードウェアに対する SBOM ユースケースの作成

出所) SBOM-a-rama における議論内容に基づき三菱総合研究所作成

2.1.3.2. FDA における取組

米国 FDA は、2018 年 10 月に市販前医療機器のサイバーセキュリティ管理内容に関するガイダンスのドラフトである「Content of Premarket Submissions for Management of Cybersecurity in Medical Devices (Draft Guidance for Industry and Food and Drug Administration Staff)」⁴³を発表した。本ドラフトは、ファームウェアを含むソフトウェアが搭載された医療機器や医療用ソフトウェアにおける市販前申請を対象としたもので、医療機器の設計・開発におけるサイバーセキュリティの取組に関して、機器資産、脅威、及び脆弱性を特定すること、機器の機能及びエンドユーザーや患者に対する脅威の影響と脆弱性を評価すること、脅威や脆弱性が悪用される可能性を評価すること、リスクレベルの判定とリスク軽減策を検討すること、残存リスク及びリスク判定基準の評価を行うことが推奨されている。FDA ガイダンス文書は、ガイダンス発行時点での FDA の考えや推奨事項を記述するものであるため、法的強制力はない。他方で、規制当局から発表されるガイダンスであるため、最終版が発表された場合には相応の影響力が存在する。本ドラフトガイダンスでは、脆弱性が存在する又は存在する可能性のある商用、オープンソース及び既成のソフトウェアコンポーネントとハードウェアコンポーネントのリストである CBOM (Cybersecurity Bill of Materials) に関する言及がなされている。具体的には、医療機器の設計において自動的に利用できる機械判読可能な CBOM の利用を推奨している。また、医療機器のサプライヤーに対しては、ユーザーによる資産の効率的な管理や脆弱性の影響把握、機器への対策管理のために CBOM をユーザーに提供することを推奨している。

関連する国際的な取組として、国際医療機器規制当局フォーラム (International Medical Device Regulators Forum: IMDRF) により取りまとめられた医療機器サイバーセキュリティガイダンス N60 「Principles and Practices for Medical Device Cybersecurity (医療機器サイバーセキュリティの原則及び実践)」⁴⁴の発表が挙げられる。本ガイダンスでは、医療機器の製造業者において SBOM を作成し、機器のオペレータに対して SBOM を配布することを奨励しているとともに、機器のオペレータにおいては、SBOM を活用して脆弱性が潜んでいる可能性があるソフトウェアの特定や、適切なリスクマネジメントを講じることを推奨している。この IMDRF によるガイダンスは 2020 年 4 月に公開されたものであるが、IMDRF の管

⁴³ <https://www.fda.gov/regulatory-information/search-fda-guidance-documents/content-premarket-submissions-management-cybersecurity-medical-devices>

⁴⁴ IMDRF、Principles and Practices for Medical Device Cybersecurity (医療機器サイバーセキュリティの原則及び実践) <https://www.mhlw.go.jp/hourei/doc/tsuchi/T200521I0040.pdf>

理委員会において当該ガイダンスに関する新業務項目延長が実施され、現在、追加事項の検討がなされている。具体的には「Medical Device Cybersecurity Deeper Dive: Legacy Devices and Transparency of Software Components Including Use of Third-Party Software（医療機器サイバーセキュリティの深掘り：レガシー機器及びサードパーティ製ソフトウェアを含むソフトウェアコンポーネントの透明性について）」に関する取組が推進されており、SBOMの観点では、SBOMの実装やサードパーティ製ソフトウェアの使用及びサポートに必要な透明性の課題に取り組むことが予定されている。新業務項目の開発は2021年2月から開始され、最終公開は2023年4月に予定されている⁴⁵。

参考として、医療機器におけるSBOMの議論は米国に限らず各国で推進されている。日本も含めた諸外国における医療機器分野のSBOM利活用における取組比較を表2-13に示す。日本では、IMDRFのガイダンスを踏まえ、厚生労働省より2021年12月に「医療機器のサイバーセキュリティの確保及び徹底に係る手引書」が公開された。この手引書ではSBOMに関する言及がなされている。なお、今後IMDRFにおける検討の動向に沿って改訂される可能性があることに留意する必要がある。

オーストラリアでは、医療機器及び体外診断用機器のサイバーセキュリティに関する要件を規定したガイダンスにおいて、当該機器のサプライヤーに対して、SBOMの作成・維持を要求している。SBOMには、サードパーティソフトウェア、開発過程が不明なソフトウェア、バージョン情報を含めることの必要性が明記されている。

カナダにおいても、医療機器におけるサイバーセキュリティに関する市販前届出要件を規定したガイダンスにおいて、医療機器サプライヤーに対して、SBOMの作成を要求している。具体的なSBOMに関する要求事項として、SBOMには各ソフトウェアコンポーネントのバージョン情報とビルド情報を含めることを求めている。また、対象となる医療機器について高度なリスク管理が必要と判断された場合、SBOMには、リスク低減・排除を目的とした機器の操作方法、バックアップ及び復元機能に関する情報、機器のソフトウェア更新方法等に関連する情報を含めるべきとしている。

⁴⁵ 関水 英正、IMDRF 活動報告会：Cybersecurity WG <https://www.pmda.go.jp/files/000243546.pdf>

表 2-13 国内外における医療機器分野の SBOM に関する取組概要

国	日本	米国	オーストラリア	カナダ
取組主体	厚生労働省	FDA（食品医薬品局）	TGA（薬品・医薬品行政局）	Health Canada カナダ保健省
SBOM 管理に関する文書	医療機器のサイバーセキュリティの確保及び徹底に係る手引書（2021 年 12 月） ⁴⁶	Content of Premarket Submissions for Management of Cybersecurity in Medical Devices (Draft)（2018 年 10 月）	Medical device cyber security guidance for industry Ver 1.1（2021 年 3 月） ⁴⁷	Guidance Document: Pre-market Requirements for Medical Device Cybersecurity（2019 年 6 月） ⁴⁸
SBOM に関する要件	- 顧客向けの文書として、注意事項等情報及び取扱説明書に加え、医療機器製品に実装されているオープンソース及び市販のソフトウェア部品（製品コンポーネント）の透明性を確保するための SBOM を含めること。 - SBOM は、患者を含む医療機器のユーザーが、その資産を効果的に管理し、医療機器及び接続されるシステムに対して識別された脆弱性の潜在的影響を理解し、医療機器の安全性及び性能を維持するための対応を可能にするものとして位置づけられる。	- 医療機器設計において、自動的に利用可能・機械判読可能な CBOM の利用が推奨される。 - 医療機器のサプライヤーにおいて、ユーザーによる資産の効率的な管理や、脆弱性の影響把握、機器への対策管理のため、CBOM をユーザーに提供することが推奨される。 - CBOM には、脆弱性の影響を受ける可能性がある商用、オープンソース、既成ソフトウェア及びハードウェアのコンポーネントを含むことが推奨される。	- 特定の医療機器に影響を及ぼす可能性のある脆弱性を監視するために、医療機器サプライヤーは、脆弱性が発見された場合のリスク評価を改善する SBOM を維持する必要がある。 - SBOM には、サードパーティソフトウェア、開発過程が不明なソフトウェア、バージョン情報を含む必要がある。	- 医療機器サプライヤーは、医療機器用ソフトウェアに含まれるすべてのコンポーネントをリスト化した CBOM を作成すること。CBOM には、各コンポーネントのバージョンとビルド情報を含むこと。 - 高度なリスク管理が必要と判断された場合、CBOM には、リスク低減・排除を目的とした機器の操作方法、バックアップ及び復元機能に関する情報、機器のソフトウェア更新方法等に関連する情報を含めるべきである。

出所）各国の取組に関する公開情報に基づき三菱総合研究所作成

⁴⁶ <https://www.mhlw.go.jp/hourei/doc/tsuchi/T211228I0070.pdf>

⁴⁷ <https://www.tga.gov.au/book-page/post-market-guidance>

⁴⁸ <https://www.canada.ca/en/health-canada/services/drugs-health-products/medical-devices/application-information/guidance-documents/cybersecurity/document.html>

2.1.3.3. DoD における取組

2022 年 1 月 26 日、米国 DoD は「Software Development and Open Source Software」と題する同省 CIO (John B. Sherman 氏) の書面を公開した⁴⁹。この書面では、既存の政府ソリューションや OSS の採用を優先する方針が示されている。書面では、OSS を採用する際の 2 つの懸念を示したうえで、書面に付属するガイドラインにおいて具体的な方針を示している。書面で示された 2 つの懸念は以下のとおりである。

- **悪意のあるコードが混入する恐れがある：**
外部でメンテナンスされているコードを用いることで、敵対者が国防総省のシステムに悪意のあるコードを入れ込む経路を作る可能性がある。
- **敵対者により悪用される恐れがある：**
DoD のために開発されたソフトウェアのコードを OSS として開示することで、重要なイノベーションの開示につながり、敵対者に対して利益をもたらす可能性がある。

前者の懸念に対しては、他の製品の調達と同様に、OSS に対しても厳格なサプライチェーンリスク管理の基準を満たすこと、脆弱性診断を通過することを要件とすることで対応可能としている。また、後者の懸念に対しては、MOSA (Modular, Open-Systems Approach) と呼ばれるオープンな構成要素を組み合わせたシステム構築手法を採用することで、リスクと切り離したシステム構築が可能であるとしている。

書面の付録では、「Guidance on Software Development and Open Source Software」として、ソフトウェア開発と OSS の取扱いに関するガイダンスが示されている。本ガイダンスの 2 章では、「プロプライエタリな製品を購入する前に、既存の政府又は OSS のソリューションを優先的に採用し、既存のソリューションが十分でない場合にのみ、非商用のソフトウェアを作成しなくてはならない。」と明記されており、前述のとおり、プロプライエタリ・ソフトウェアと比較して OSS の採用を優先する旨が明確に打ち出されている。そのうえで、OSS のサプライチェーンリスク管理にあたっては、OSS 特有の課題があり、それぞれの課題に対して適切な対応策を講じる必要があると明記されている。例えば、OSS を作成・維持するビジネスモデルは多岐にわたるため、各 OSS が適切にサポートされるか、リスクがどのように軽減可能であるかを考慮する必要があるとしているほか、各 OSS がアクティブであるか、最後に修正されたのはいつか、誰が貢献して管理されているか等の「健全性」を評価すべきとしている。ほかにも、OSS のソースを慎重に検討し、そのソースの信頼性を適切に判断すべきであり、OSS の管理にあたっては、ゼロトラストのアプローチに従い、「Never trust, always verify」のアプローチが重要であることが示されている。

2.2. 企業や業界団体、OSS コミュニティにおけるソフトウェア管理の取組動向の調査

本調査項目では、企業等において自社製品・サービスの一部として利活用するソフトウェアに既存の脆弱性が存在せず、安全なソフトウェアであることを確認するための取組や、当該目的を達成するための製品・サービスの最近の動向について調査した。

⁴⁹ <https://dodcio.defense.gov/portals/0/documents/library/softwaredev-opensource.pdf>

2.2.1. 公開情報調査結果

2.2.1.1. Software ISAC における取組

コンピュータソフトウェア協会(現:ソフトウェア協会)内の組織として発足した Software ISAC は、ソフトウェア・サプライチェーンの強靱化に関する研究を行い、安心・安全な日本への貢献を行う開発者のための情報交換基盤を提供することを目的に活動している。2021 年 4 月 28 日、Software ISAC は、ソフトウェア開発の管理者や責任者に向け、ソフトウェア管理の在り方をまとめた「ソフトウェアの安全性を意識した管理体制 (ver.1.0)」を公開した⁵⁰。文書では、ソフトウェアにおける課題と、ソフトウェア管理体制の構築に向けた以下のポイントが示されている。

① 契約状況の把握：

契約内容、種類、契約期間、損害賠償上限、自動更新の有無、再委託可否等を確認する。

② プロジェクトの把握：

開発内容、プロジェクト体制、各担当者の作業内容・開発環境等を確認する。

③ 開発環境や開発に使うソフトウェア情報の把握：

開発環境、使用した OSS、SBOM コンポーネント、開発環境におけるセキュリティ対策等を確認する。

④ 関係者のスキルやリテラシーの把握：

特に開発に直結するステークホルダーについて、スキルやリテラシーを把握する。

⑤ 管理体制のメンバーの決定と把握：

ソフトウェアの脆弱性が生じた場合の管理体制の責任者、外部との連携窓口を明確にする。

⑥ ステークホルダー（利害関係者）全体の把握：

ソフトウェア開発の利害関係者をリスト化する。

⑦ ツールやシートを使った管理：

①～④の情報を管理するツールを書き出し、維持・管理するための必要な情報を整理する。

2.2.1.2. Linux Foundation における取組

2000 年に設立された Linux Foundation は、Linux OS を中心としたオープンソース・プロジェクトに関する持続可能なエコシステムを構築し、技術の発展や商業的な活用促進を目的とした非営利の技術コンソーシアムである。OSS のセキュリティ向上のための取組も推進しており、特に、2020 年 8 月に Open Source Security Foundation (OpenSSF) を設立して以降、業界横断的な取組に注力している。

2.2.1.2.1. Open Source Management & Strategy Program

2021 年 1 月、組織内の管理職や技術担当者にオープンソースのベストプラクティスを紹介する新たなトレーニングプログラム「Open Source Management & Strategy（オープンソー

⁵⁰ https://www.saj.or.jp/NEWS/activity/softwareisac/210428_management.html

スの管理と戦略)」プログラムを発表した⁵¹。プログラムは7つのコースで構成されており、経営者、管理者、ソフトウェア開発者、エンジニア等が、組織内で効果的なオープンソースプラクティスを構築するための基本的な概念を理解・説明できるように設計されている。具体的な7つのコースは以下のとおりである。

- LFC202 - イントロダクション: オープンソースとオープンスタンダードの基本的なコンポーネントについて
- LFC203 - オープンソースビジネス戦略: 様々なオープンソース・ビジネスモデルと、それぞれの実践的な戦略や方針を策定する方法
- LFC204 - 効果的なオープンソース・プログラム管理: 効果的な OSPO (オープンソース・プログラム・オフィス) を構築する方法と、それを運営するために必要な様々なタイプの役割と責任
- LFC205 - オープンソース開発プラクティス: 健全なオープンソース・プロジェクトにおける継続的インテグレーションとテストの役割
- LFC206 - オープンソース・コンプライアンス・プログラム: 効果的なオープンソース・ライセンス・コンプライアンスの重要性、オープンソースを安全で効果的に消費できるようにするためのプログラムとプロセスの構築方法
- LFC207 - オープンソース・プロジェクトとの効果的なコラボレーション: アップストリームのオープンソース プロジェクトと効果的に連携する方法、プロジェクト・コミュニティとの作業から最大の利益を得る方法
- LFC208 - オープンソース・プロジェクトの作成: 新しいオープンソース・プロジェクトを作成する理由と価値、新しいプロジェクトを立ち上げるために必要な法的プロセス、業務プロセス、開発プロセス

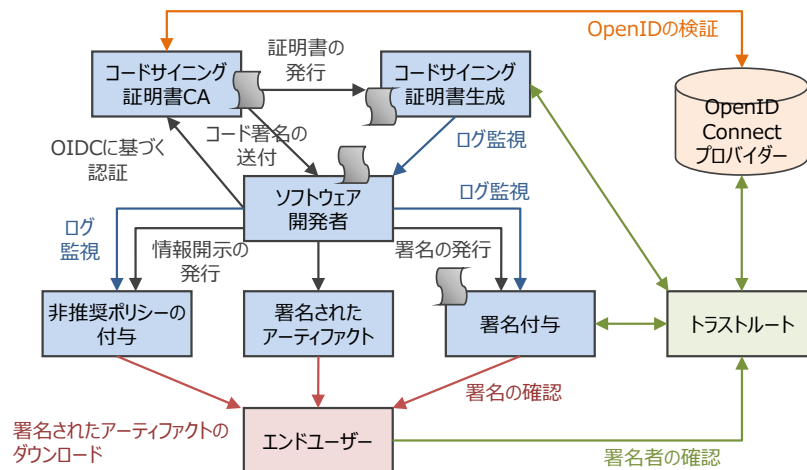
受講者は年間 499 ドルを支払うことで7つのコースすべてに無制限にアクセスでき、修了時には修了証が取得できる。

2.2.1.2.2. sigstore Project

2021 年 3 月、Linux Foundation は Red Hat や Google 等とともに、オープンソース・ソフトウェアの署名サービスである「sigstore」プロジェクトの活動開始を発表した⁵²。sigstore 署名サービスを活用することで、透明性が確保された暗号化ソフトウェア署名を容易に導入することができ、ソフトウェア・サプライチェーンのセキュリティ向上に寄与するとしている。sigstore のアーキテクチャを図 2-8 に示す。sigstore は、ソフトウェア開発者がコンテナイメージやバイナリファイル等のソフトウェア・アーティファクトに安全に署名できる仕組みであり、署名資料は、改ざんが防止された形で公開ログに保存される。すべてのソフトウェア開発者が sigstore を無償で利用できる。

⁵¹ <https://www.linuxfoundation.jp/press-release/2021/01/open-source-management-strategy-training-program-launched-by-the-linux-foundation/>

⁵² <https://www.sigstore.dev/>



出所) sigstore に関する公開情報に基づき三菱総合研究所作成

図 2-8 sigstore のアーキテクチャ

2.2.1.2.3. Security Scorecards 2.0

2021 年 6 月、OpenSSF と Google は、GitHub レポジトリのオープンソース・プロジェクトのセキュリティをスコア化するツール「Security Scorecards 2.0」を発表した⁵³。本プロジェクトは 2020 年 11 月に発表された「Scorecards」のバージョン 2.0 の位置づけであり、2.0 ではチェック項目が新たに追加された。「Security Scorecards 2.0」を使用することで、オープンソース・プロジェクトのセキュリティに関する 17 の項目に対し、自動でセキュリティチェックが行われ、セキュリティ対策状況の「合格／不合格」の判定と「0～10 の信頼度」（10 に近いほど、判定結果を信頼できる）が出力される。「Security Scorecards 2.0」の 17 の評価項目は以下のとおりである。

表 2-14 「Security Scorecards 2.0」の評価項目

#	項目	説明
1	Active	90 日以内にプロジェクトはコミットされているか。
2	Automatic Dependency Update	プロジェクトの依存関係が自動更新されるか。
3	Binary Artifacts	ソースコードにバイナリアーティファクトがあるか。
4	Branch Protection	意図的に悪意のあるコードを挿入できないようブランチ保護されているか。
5	CI Tests	プルリクエストを実行する前にテストを実行しているか。
6	CII Best Practices	CII ベストプラクティスバッジ ⁵⁴ を有しているか。
7	Code Review	プルリクエストを実行する前にコードレビューを要求しているか。

⁵³ <https://github.com/ossf/scorecard>

⁵⁴ CII とは Core Infrastructure Initiative の略であり、Linux Foundation の一プロジェクトのことである。CII ベストプラクティスバッジとは、Linux Foundation が定めたオープンソース・プロジェクトのセキュリティ管理に関するベストプラクティスに従っていることを開発者自ら示すバッジのことで、以下の Web サイトで概要が記載されている。<https://github.com/coreinfrastructure/best-practices-badge>

#	項目	説明
8	Contributors	プロジェクトに複数の参加企業がいるか。
9	Fuzzing	ファuzzingテストを実施しているか。
10	Frozen Deps	プロジェクトは依存関係を宣言して固定されているか。
11	Packaging	第三者がインストール/ダウンロードできるパッケージとして公開されているか。
12	SAST	SAST ツールで静的コード分析を実行しているか。
13	Security Policy	プロジェクトにセキュリティポリシーはあるか。
14	Signed Releases	アーティファクトは署名されているか。
15	Signed Tags	GitHub の Tag は署名されているか。
16	Token Permissions	GitHub のアクセス権はデフォルトで読み取り専用であるか。
17	Vulnerabilities	未修正の脆弱性があるか。（脆弱性の評価は Open Source Vulnerabilities データベース（第 2.2.1.3.1 項参照）を基に実施される）

出所) Security Scorecards 2.0 に基づき三菱総合研究所作成

2.2.1.2.4. Secure Open Source (SOS) Rewards Pilot Program

2021 年 10 月、Linux Foundation は OSS のセキュリティ向上のための報酬プログラムである「Secure Open Source (SOS) Rewards」のパイロットプログラムの開始を発表した⁵⁵。本プログラムは、重要な OSS のセキュリティ強化に貢献する開発者に対して報酬を付与するプログラムであり、Linux Foundation が運営、Google がスポンサーの位置づけで参画している。現状で発表されている報酬の合計は 100 万ドルである。個別の報酬の目安として、複雑で影響度が高い改善に対しては 1 万ドル、適度に複雑で改善によりセキュリティメリットが享受できる改善は 5,000～1 万ドル、適度な影響度がある改善は 1,000～5,000 ドル、小規模な改善だが多少のセキュリティメリットがある改善は 505 ドルを目安としている。

本プログラムの対象となる OSS の範囲については明確に定義されていないものの、NIST により公開された「Definition of Critical Software Under Executive Order (EO) 14028」（第 2.1.1.2 項参照）の範囲を考慮としている。また、報酬の対象となる改善貢献内容について、ソフトウェア・サプライチェーンに関するセキュリティ、ソフトウェア・アーティファクトへの署名の適用、「Security Scorecards 2.0」の評価結果が向上するような改善を対象としている。

2.2.1.2.5. Alpha-Omega Project

2022 年 2 月 1 日、Linux Foundation 傘下の OpenSSF は、オープンソース・ソフトウェアのサプライチェーン課題を改善し、オープンソース・ソフトウェアの安全性を高めるための

⁵⁵ <https://sos.dev/>

プロジェクトである Alpha-Omega Project の開始を発表した⁵⁶。このプロジェクトは、2022 年 1 月にホワイトハウスで開催された官民会議（第 2.1.1.9 項参照）での議論に基づき発表されたもので、OSS における未知の脆弱性を探索・修正により、グローバルサプライチェーンでのセキュリティ向上に寄与することを目的としたプロジェクトである。

本プロジェクトは、Alpha プロジェクトと Omega プロジェクトという 2 つから構成される。前者は最も重要と評価されたオープンソース・プロジェクトに協力し、セキュリティ体制の改善を個別に支援するプロジェクトである。OpenSSF のワーキンググループや専門家によって評価がなされ、採択されたプロジェクトに対しては、脅威分析、セキュリティ診断、ソースコード解析、脆弱性の対応等の支援を受けることができる。後者の Omega プロジェクトでは、1 万件以上のオープンソース・プロジェクトを対象にセキュリティ対策を支援することが発表されている。本プロジェクトの実施にあたって、Microsoft 及び Google が共同で 500 万ドルを拠出するとともに、専門家を派遣することを発表している。

2.2.1.3. Google における取組

2.2.1.3.1. Open Source Vulnerabilities (OSV) Project

2021 年 2 月、Google は OSS の脆弱性をバージョンごとにデータベース化する Open Source Vulnerabilities (OSV) プロジェクトを開始した⁵⁷。OSV データベースの Web サイト上で OSS コンポーネントの名称を入力すると、依存関係のあるライセンスや、依存関係のあるソフトウェアが一覧表示やグラフ表示で確認できる。OSS の開発者がこのデータベースを活用することで脆弱性の報告が容易になるほか、OSS 利用者においては、脆弱性情報に関する正確な情報を得ることができ、利用するソフトウェアの管理と対応を迅速かつ容易に実施することが可能となる。OSV プロジェクトの詳細については第 4.1.5.1 項に記載しているため、適宜参照のこと。

2.2.1.3.2. Open Source Insight Project

2021 年 6 月、Google は OSS の依存関係を可視化する Open Source Insights プロジェクトを発表した⁵⁸。本サービスでは、OSS の脆弱性に関する情報がデータベース化されており、それぞれの脆弱性に関して、対応する CVE 番号や情報ソース、脆弱性の影響を受けるバージョン、対策情報等を確認することができる。このサービスを活用して OSS の依存関係を確認することで、OSS のセキュリティリスク等の判断を行うことができる。

2.2.1.4. 欧州 ENISA における取組

2021 年 7 月、ENISA は 24 件のサプライチェーン攻撃事例に関する調査・分析結果を示すレポート「Threat Landscape for Supply Chain Attacks」を公開した⁵⁹。本レポートでは 2020 年

⁵⁶ <https://openssf.org/press-release/2022/02/01/openssf-announces-the-alpha-omega-project-to-improve-software-supply-chain-security-for-10000-oss-projects/>

⁵⁷ <https://osv.dev>

⁵⁸ <https://deps.dev/>

⁵⁹ <https://www.enisa.europa.eu/publications/threat-landscape-for-supply-chain-attacks>

1月から2021年7月にかけて発生した24件のサプライチェーン攻撃を分類整理し、それぞれの攻撃を分析している。具体的な攻撃事例として、SolarWindsの事例、Codecovの事例、Kaseyaの事例、Apple Xcodeの事例等が含まれ、ENISAにより提案されたサプライチェーン攻撃の分類法（表2-15）に基づき、各攻撃の特徴を分析した。

表 2-15 ENISA により提案されたサプライチェーン攻撃の分類法

サプライヤーに対する サプライチェーン攻撃		顧客に対する サプライチェーン攻撃	
用いられる攻撃技術	攻撃対象	用いられる攻撃技術	攻撃対象
● マルウェア感染 ● ソーシャル・エンジニアリング ● ブルートフォース ● ソフトウェア脆弱性の悪用 ● コンフィグレーション脆弱性の悪用 ● OSINT 情報	● 既存のソフトウェア ● ライブラリ ● コード ● コンフィグレーション ● データ ● プロセス ● ハードウェア ● 人員 ● サプライヤー（組織）	● 信頼関係の悪用 ● Web 閲覧による感染 ● フィッシング ● マルウェア感染 ● 物理的攻撃、改変 ● 偽造、模倣品	● データ ● 個人情報 ● 知的財産 ● ソフトウェア ● プロセス ● 処理能力 ● 金銭 ● 人員

出所）ENISA 文書に基づき三菱総合研究所作成

また、ENISA は同月、調査・分析結果を踏まえ、サプライチェーン攻撃に関する現状説明やサプライヤー及び顧客（調達者）が実施すべき推奨事項をまとめたプレス発表を行った⁶⁰。主な調査・分析結果として ENISA は以下の点を挙げている。

- 50%の攻撃は有名な APT グループにより実施された
- 62%の攻撃において、マルウェアが攻撃技術として採用
- 58%の攻撃がデータ（個人情報、知財等の顧客の情報）へのアクセスを目的としていた
- サプライヤーに対する攻撃の 66%は、サプライヤーのコードを対象とした
- 顧客に対する攻撃の 62%は、顧客とサプライヤーとの信頼関係を悪用した

ENISA は調査・分析結果を踏まえ、サプライヤーが実施すべき推奨事項や顧客が実施すべき推奨事項をそれぞれ整理した。サプライヤーが実施すべき推奨事項として、製品、コンポーネント、サービスの設計、開発、製造、提供に使用されるインフラが、サイバーセキュリティのプラクティスに従っていることを確認することや、一般的に認められた製品開発プロセスに整合する製品開発、保守、サポートのプロセスを実施すること等を挙げた。また、顧客が実施すべき推奨事項として、サプライヤーとサービスプロバイダーを特定し、文書化することや、様々なタイプのサプライヤーやサービスのリスク基準を定義すること等を挙

⁶⁰ <https://www.enisa.europa.eu/news/enisa-news/understanding-the-increase-in-supply-chain-security-attacks>

げた。

2.2.1.5. 英国における取組

2021 年 12 月、英国より新たな「国家サイバー戦略」が公開された⁶¹。この戦略では優先的に実行すべき 5 つの観点（図 2-9）を戦略の柱として、2025 年までの達成目標と達成のための具体的な行動がまとめられている。

1	サイバーエコシステムの強化	- 産業界や諸外国との連携を拡大し、社会全体でのネットワークを強化する。 - サイバー人材を育成することにより、国家のサイバースキルを強化・拡大する。 - 政府や経済のニーズを満たす質の高いサービスを提供するため、持続可能で国際的に競争力のあるサイバー機関の成長を促進する。 等
2	サイバー攻撃に対する理解・対策・対応力のある国家の構築	- サイバースキルの理解を深め、サイバー攻撃に対する効果的な行動を推進する。 - サイバースキルを管理し、サイバー攻撃を防止するためのサポートやガイダンスを提供する。 - サイバー攻撃に対応し、被害を最小限にするためのサポートを提供する。 等
3	サイバー空間の競争力維持や保護に不可欠な技術の先導	- サイバー空間の競争力を維持するため、ITの発展を予測・評価・行動する能力を向上する。 - クラウドやソフトウェア等のデジタルサービスに対してサイバーセキュリティ基準への準拠を求め、サイバー脅威から消費者を保護する。 - 次世代のコネクテッド技術やインフラを構築することでサイバーセキュリティリスクを軽減し、ユーザーへの多様なサービス提供を実現する。 - サイバーセキュリティを確保しながら、ITを通じて英国の戦略的優位性を高めるため、デジタル技術の標準化に向けた議論を先導する。 等
4	グローバルでのリーダーシップや影響力の向上、及び安全かつ豊かで開かれた国際秩序の推進	- 国際的パートナーとともに、サイバー攻撃者に対する対策・対応力を強化する。 - サイバー空間におけるグローバルガバナンスを形成し、オープンなサイバー空間の構築を促進する。 - サイバー能力や専門知識を国外へ提供し、戦略的優位性を高め、より広範な外交政策と国家繁栄のための利益を促進する。 等
5	サイバー空間におけるあらゆる攻撃の抑止・検出	- サイバー攻撃に関する情報を検出・調査・共有する。 - 法整備、潜在的なサイバー攻撃者への抑止プログラムの構築により、サイバー攻撃を抑止する。 - サイバー空間での活動を通じ、安全保障の実現や重大犯罪の防止・検出を行う。 等

出所）英国「国家サイバー戦略」に基づき三菱総合研究所作成

図 2-9 英国国家サイバー戦略の 5 つの柱の概要

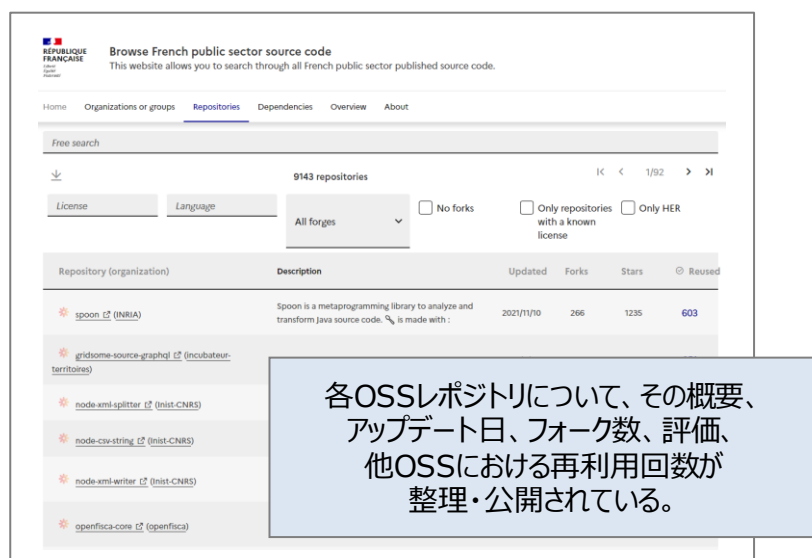
英国はこの戦略を通じて、サイバー空間における英国の経済的・戦略的な強みを強化し、社会のあらゆる部分に役割を果たすとしている。ソフトウェアセキュリティに関連する戦略として、クラウドやソフトウェア等のデジタルサービスの主要プロバイダーに対してサイバーセキュリティ基準への準拠を求めるとともに、サイバー脅威から消費者を保護することに注力している。また、デジタル・サプライチェーンのセキュリティ確保のために、国際的な取組をリードする旨が明記されている。

2.2.1.6. フランス Etalab における取組

2021 年 11 月、フランスデジタル省庁間総局（DINUM : Direction Interministérielle du

⁶¹ <https://www.gov.uk/government/publications/national-cyber-strategy-2022/national-cyber-security-strategy-2022>

Numérique) の一部門である Etalab は、フランスの公的機関が公開している OSS レポジトリを集めた Web サイトの公開を開始した⁶²。当該サイトは 2019 年 10 月の政令 (Décret n° 2019-1088 du 25 octobre 2019 relatif au système d'information et de communication de l'Etat et à la direction interministérielle du numérique) ⁶³に基づき公的機関における OSS の活動を促進する目的で構築・公開され、2022 年 1 月時点で 1,000 組織を超える公的機関の 9,000 を超える OSS レポジトリ情報が公開されている。各 OSS レポジトリについて、公開している公的機関のプラットフォームへのリンクが用意されているほか、対象 OSS で採用されているライセンスの割合や、対象 OSS で再利用している他の OSS 等の統計情報も確認可能である。なお、米国においても、米国政府機関の OSS レポジトリを公開するサイト⁶⁴が 2016 年 11 月より運用されている。



出所) <https://code.gouv.fr/>

図 2-10 フランスの公的機関による OSS レポジトリを集めた Web サイトの概要

2.2.2. ヒアリング調査結果

国内企業におけるソフトウェア管理に関する取組状況をヒアリング調査で確認した。本調査では、ソフトウェアタスクフォースの委員の所属企業であるトヨタ自動車株式会社、日本電気株式会社 (NEC)、富士通株式会社に対してヒアリングを実施した。ヒアリングでは、各企業における取組の概要だけでなく、各企業が抱えているソフトウェア管理における課題についても聴取した。共通的な課題として、SBOM 共有のためはフォーマットや部品粒度等を揃える (要件化する) 必要性があること、委託先や OSS の再帰的な利用 (OSS による他の OSS の再利用等) を含めて SBOM 管理を徹底する場合のコストが大きい点などが挙げられた。

⁶² <https://code.gouv.fr/>

⁶³ 情報通信システムの分野に関する行政省庁の連携・調整を進めるデジタル省庁間総局 (DINUM) を設置し、情報通信システムに関する国全体としての管轄について定めた政令。DINUM の所掌任務の一つとして、公的機関の監督化にあるデータの利活用を最大限促進することを規定しており、この規定に基づき OSS レポジトリを集めた上記 Web サイトが公開された。

⁶⁴ <https://code.gov/>

2.2.2.1. トヨタ自動車株式会社における取組

トヨタ自動車では、著作権法やライセンス規約への準拠のため、OSS を使用している一部の ECU（エンジンコントロールユニット）について、SBOM 管理を実施しているとのことであった。また、部品サプライヤーの理解を得て、部品情報の提供を契約で規定し、脆弱性の管理にも活用している。トヨタ自動車が抱えている課題として、事業者間で SBOM を共有するためには、部品の粒度を揃えることが必要である点や、ライセンス準拠等に関する部品サプライヤーの責任を明確化しているものの、Tier 2 以下のサプライヤーについては、部品情報が網羅されているか確認することが難しい点が挙げられた。

2.2.2.2. 日本電気株式会社（NEC）における取組

NEC では、OSS について SBOM 相当の部品管理を実施しているとのことであった。また、委託先に対する SBOM の要件化は現状で実施できていないが、OSS の利用申告は規定しているとのことであった。NEC が抱えている課題として、サプライチェーンで部品粒度を整合化しなければ、かえって非効率となることが挙げられた。また、スニペットの検出精度等、既存の SBOM 生成ツールには課題があったことから、十分に比較検証を行ったうえでツールを選定したとのことであった。加えて、OSS の再帰的な利用も含めた厳格な検査はコスト的に困難であるものの、改善について継続的に検討を実施していること、委託先開発部分や OSS 利用も含めた完全な管理は困難であるが、最終製品ベンダーとしての責任において適切な対応を模索中であることが挙げられた。

2.2.2.3. 富士通株式会社における取組

富士通では、脆弱性管理、ライセンス管理を目的に SBOM を導入しており。厳格に SBOM を管理運用する場合はコストが非常に高額となるため、ビジネスインパクトに応じて簡約して管理しているとのことであった。また、直接の取引相手とは SBOM を共有しているが、サプライチェーン全体ではほとんど共有ができていない状況とのことであった。富士通が抱えている課題として、再帰的な OSS 利用も含めた解析も可能な限り実施しているものの、様々な形態の製品・サービスを社内・社外向けに提供しており、それらすべてのビジネスタイプにおいて徹底することはコスト的に困難であることが挙げられた。また、ビジネスタイプによるフォーマットやプロセスの違いから、サプライチェーン全体で特定の SBOM フォーマットを要件化することができず、SBOM 共有の障害となっているとの課題も挙げられた。加えて、契約において SBOM の瑕疵責任を規定したとしても最終的な責任は最終製品ベンダーにあるため、レピュテーションリスクを考慮すれば、SBOM の検査を受入側でやらざるを得ないとの意見も挙げられた。

2.3. OSS を含むソフトウェアの安全な利活用に関する基礎調査

本調査項目では、OSS を含むソフトウェアの安全な利活用の必要性について理解を助けるための基礎資料として、ソフトウェアの安全を脅かす要因やその対処方法、OSS 関連の最近の動向等、特にソフトウェアの脆弱性事例やインシデント事例について、公開情報を中心に収集した。調査結果を表 2-16 に示す。そして、調査結果について、ソフトウェアタスクフォースで紹介することを前提として取りまとめた。本事業では、表 2-16 に示す事例の

うち、No. 1、No. 3、No. 4、No. 13 の事例についてソフトウェアタスクフォースの資料として取りまとめた。

表 2-16 ソフトウェアに関連するインシデント・脆弱性事例

#	名称	事例報告日	概要
1	ソフトウェア開発のテスト支援ツール「Codecov」にバックドア ⁶⁵	2021 年 4 月 15 日	2021 年 4 月、Codecov に含まれる Bash Uploader Script が不正に書き換えられ、利用者の資格情報等が不正に外部に送信され、ソースコードなどの機密情報が漏えいするリスクがあることが利用者指摘により発覚。 Bash Uploader Script が書き換えられていた可能性があるのは、2021 年 1 月 31 日～2021 年 4 月 1 日の期間であり、国内でも不正取得された資格情報等を悪用したソースコードや顧客情報の漏えい事案が確認されている。
2	BIND 9 の脆弱性 ⁶⁶	2021 年 4 月 30 日	ISC BIND 9 の脆弱性が発表された。攻撃者が脆弱性を悪用することで、DNS サーバープロセスを異常終了される可能性がある。対策のために、パッチバージョンをアップデートするよう求めた。
3	Windows 印刷スプーラーサービスの脆弱性：PrintNightmare ⁶⁷	2021 年 7 月 1 日	Windows 印刷スプーラーサービスの脆弱性「PrintNightmare」が発表された。攻撃者はリモートコード実行の脆弱性を悪用することで、SYSTEM 権限で任意のコードが実行され、データの変更や削除、アカウントの作成等が実行される可能性がある。Microsoft は、対策のために印刷スプーラーサービスの無効などによるリモート印刷を無効にするよう推奨した。

⁶⁵ Codecov, Bash Uploader Security Update <https://about.codecov.io/security-update/>

⁶⁶ JPCERT/CC, ISC BIND 9 の複数の脆弱性に関する注意喚起 <https://www.jpcert.or.jp/at/2021/at210021.html>, JPCERT/CC, ISC BIND 9 の脆弱性 (CVE-2021-25218) に関する注意喚起 <https://www.jpcert.or.jp/at/2021/at210035.html>

⁶⁷ Microsoft, Windows 印刷スプーラーのリモートでコードが実行される脆弱性 <https://msrc.microsoft.com/update-guide/vulnerability/CVE-2021-34527>, JVN, Microsoft Windows の印刷スプーラーにリモートコード実行の脆弱性 <https://jvn.jp/vu/JVNVU96262037/>

#	名称	事例報告日	概要
4	IT 管理サービス 「Kaseya VSA」の脆弱性を利用したサプライチェーンランサムウェア攻撃が発生 ⁶⁸	2021 年 7 月 2 日	Kaseya IT 管理サービスの「VSA」の脆弱性を悪用し、サプライチェーンランサムウェア攻撃が実行されたと発表した。Kaseya は、対策のために「VSA」を停止するよう求めた。
5	Linux Kernel の脆弱性 ⁶⁹	2021 年 7 月 20 日	Qualys により、Linux Kernel の脆弱性が発表された。攻撃者が脆弱性を悪用することで、権限昇格させ、任意のコードが実行される可能性がある。対策のために、OS のアップデートを実施することが求められた。
6	PowerDNS 権威サーバーの脆弱性 ⁷⁰	2021 年 7 月 22 日	2021 年 7 月、OSS の DNS サーバーである PowerDNS Authoritative サーバーが特定のクエリを受け取ると、サービス不能状態になる脆弱性が発表された。対策のために、バージョンアップを求めた。
7	BlackBerry QNX Real Time OS (QNX RTOS) の脆弱性： BadAlloc ⁷¹	2021 年 8 月 17 日	CISA より、BlackBerry の QNX RTOS の脆弱性「BadAlloc」が発表された。攻撃者が脆弱性を悪用することで、任意のコードが実行され、サービス妨害状態になる可能性がある。CISA はパッチを適用することを求めた。なお、BlackBerry は同社の製品が脆弱性の影響を受けることを否定している。
8	Microsoft Exchange Server の脆弱性： ProxyShell ⁷²	2021 年 8 月 21 日	Microsoft Exchange Server の脆弱性「ProxyShell」が発表された。攻撃者が脆弱性を悪用することで、認証無しに任意のコードが実行される可能性がある。Microsoft は、対策のために最新のセキュリティ更新プログラムを適用することを求めた。

⁶⁸ Kaseya, Updates Regarding VSA Security Incident <https://www.cloudgate.jp/security-news/kaseya-product-ransomware-attack-measures.html>, Piyolog, 多数の組織がランサムウェアに感染したサプライチェーン攻撃についてまとめてみた <https://piyolog.hatenadiary.jp/entry/2021/07/12/054229>

⁶⁹ Red Hat Customer Portal, RHSB-2021-006 Long path name in mountpoint flaws in the kernel and systemd (CVE-2021-33909, CVE-2021-33910) <https://access.redhat.com/security/vulnerabilities/RHSB-2021-006>

⁷⁰ PowerDNS, PowerDNS Security Advisory 2021-01: Specific query crashes Authoritative Server <https://doc.powerdns.com/authoritative/security-advisories/powerdns-advisory-2021-01.html>

⁷¹ CISA, ICS Advisory (ICSA-21-119-04) <https://us-cert.cisa.gov/ics/advisories/icsa-21-119-04>

⁷² Sophos News, Microsoft Exchange における ProxyShell の脆弱性と対策 <https://news.sophos.com/ja-jp/2021/09/06/proxyshell-vulnerabilities-in-microsoft-exchange-what-to-do-jp/>

#	名称	事例報告日	概要
9	VPN 機器の 認証情報流出 ⁷³	2021 年 9 月 9 日	FortiGate SSL-VPN 機器の認証情報 87,000 件が公開されたことが発表された。攻撃者が当該 VPN 機器の脆弱性を悪用することで認証情報が取得された。認証情報等が悪用されることで容易に侵入される恐れがある。なお、悪用された脆弱性は 2019 年 5 月に公開されたもので、2020 年 11 月にも同様の認証情報 50,000 件が公開されており、注意喚起がなされていた。
10	Apple のメッ セージアプリ iMessage の脆弱性を利用したゼ ロクリック 攻撃 ⁷⁴	2021 年 9 月 13 日	セキュリティ機関 The Citizen Lab により、Apple のメッセージアプリ iMessage でゼロデイの脆弱性を確認したと発表した。脆弱性は 2021 年 2 月から悪用されていたと考えられており、Apple は脆弱性に対処した OS を緊急更新した。
11	CMS 製品 Movable Type におけ る OS コマン ド・インジ ェクション の脆弱性 ⁷⁵	2021 年 10 月 20 日	2021 年 11 月、Six Apart は、コンテンツ管理システム（CMS）製品 Movable Type の XMLRPC API において、OS コマンド・インジェクションの脆弱性があることを発表した。この脆弱性が悪用されることで、遠隔の第三者によって任意の OS コマンドが実行される可能性がある。IPA は、2021 年 11 月 5 日に本脆弱性を悪用した攻撃が確認されていると発表し、可能な限り早急に対策を実施するよう注意喚起を行った ⁷⁶ 。
12	UEM 製品 ManageEngin e におけるゼ ロデイ脆弱 性 ⁷⁷	2021 年 12 月 3 日	2021 年 12 月 3 日、ZOHO は統合エンドポイント管理（UEM）製品 ManageEngine において遠隔からの任意コード実行の脆弱性が存在すると発表した。また、CISA 及び FBI は 2021 年 12 月 17 日、同製品を標的とした持続的標的型攻撃（APT）が積極的に実施されているとして、警告を行った ⁷⁸ 。

⁷³ Fortinet, 悪意のあるアクターが FortiGate SSL-VPN の認証情報を公開

<https://www.fortinet.com/jp/blog/psirt-blogs/malicious-actor-discloses-fortigate-ssl-vpn-credentials>

⁷⁴ CISA, Apple Releases Security Updates to Address CVE-2021-30858 and CVE-2021-30860 <https://us-cert.cisa.gov/ncas/current-activity/2021/09/13/apple-releases-security-updates-address-cve-2021-30858-and-cve-2021-30860>

⁷⁵ Movable Type, [重要] Movable Type 7 r.5005 / Movable Type 6.8.5 / Movable Type Premium 1.49 の提供を開始（セキュリティアップデート） <https://www.sixapart.jp/movabletype/news/2021/12/16-1400.html>

⁷⁶ IPA, 更新：「Movable Type」の XMLRPC API における OS コマンド・インジェクションの脆弱性について（JVN#41119755） <https://www.ipa.go.jp/security/ciadr/vul/20211020-jvn.html>

⁷⁷ ManageEngine Desktop Central, CVE-2021-44515: Security Advisory <https://www.manageengine.com/products/desktop-central/cve-2021-44515-authentication-bypass-filter-configuration.html>

⁷⁸ CISA, Alert (AA21-259A) <https://www.cisa.gov/uscert/ncas/alerts/aa21-259a>

#	名称	事例報告日	概要
13	Apache Log4j の脆弱性： Log4Shell (CVE-2021- 44228 等) ⁷⁹	2021 年 12 月 10 日	2021 年 12 月、Java ベースのオープンソースログ出力ライブラリ Apache Log4j における任意コード実行の脆弱性が発表された。当該脆弱性は Log4j の JNDI Lookup 機能に起因するもので、Log4Shell とも呼ばれる。この脆弱性を利用することで、Log4j が動作するアプリケーションに対して外部からの任意コード実行が可能となり、情報漏えいやマルウェア感染等の被害に繋がる恐れがある。脆弱性の公表を受けて、NISC から注意喚起を発出したほか ⁸⁰ 、前後して IPA ⁸¹ 、JPCERT/CC ⁸² からも注意喚起がなされた。また、米国でも CISA 長官より声明を発表された ⁸³ ほか、12 月 13 日には脆弱性対応のためのガイダンスを発表され、連邦政府機関に対して 12 月 24 日までに「Apache Log4j」の脆弱性とそのほか 3 件の脆弱性にパッチを適用するよう命じた ⁸⁴ 。関連して 12 月 17 日には、連邦政府機関に対して、ネットワークに接続されているアセットに存在する「Apache Log4j」の脆弱性をただちに修正することを命じる追加の緊急指令を発表した ⁸⁵ 。そして 12 月 22 日には、米国や英国などをはじめとする政府のセキュリティ機関での共同声明を発表した ⁸⁶ 。

2.4. 米国 CISA との勉強会の開催

ソフトウェア管理に関する動向の調査や米国との意識合わせの目的で、米国 CISA や国内関係組織が参画した勉強会（「第 6 回ソフトウェアの利活用におけるセキュリティ確保に関する勉強会」）を 2021 年 12 月 9 日にオンラインで開催した。本勉強会では、まず経済産業省より、経済産業省における産業サイバーセキュリティ政策とソフトウェアタスクフォースの取組について発表したのち、MRI より国内の SBOM 実証の実施状況について発表を行った。その後、CISA の Allan Friedman 博士より米国の取組について発表をいただいた後、

⁷⁹ Apache, Apache Log4j Security Vulnerabilities <https://logging.apache.org/log4j/2.x/security.html>

⁸⁰ NISC, ApacheLog4j の脆弱性(CVE-2021-44228)に関する注意喚起
https://www.nisc.go.jp/press/pdf/20211213NISC_press.pdf

⁸¹ IPA, Apache Log4j の脆弱性対策について(CVE-2021-44228)
<https://www.ipa.go.jp/security/ciadr/vul/alert20211213.html>

⁸² JPCERT/CC, Apache Log4j の任意のコード実行の脆弱性 (CVE-2021-44228) に関する注意喚起
<https://www.jpcert.or.jp/at/2021/at210050.html>

⁸³ CISA, STATEMENT FROM CISA DIRECTOR EASTERLY ON “LOG4J” VULNERABILITY
<https://www.cisa.gov/news/2021/12/11/statement-cisa-director-easterly-log4j-vulnerability>

⁸⁴ CISA, Apache Log4j Vulnerability Guidance <https://www.cisa.gov/uscrt/apache-log4j-vulnerability-guidance>

⁸⁵ CISA, EMERGENCY DIRECTIVE 22-02 MITIGATE APACHE LOG4J VULNERABILITY
<https://www.cisa.gov/emergency-directive-22-02>

⁸⁶ CISA, Alert (AA21-356A) <https://www.cisa.gov/uscrt/ncas/alerts/aa21-356a>

国内関係組織も含めた自由討議の時間を設けた。

まず、CISA の Allan Friedman 博士による発表では、米国におけるこれまでの SBOM に関する取組について発表がなされた。これまで NTIA を中心に議論されてきた取組のほか、米国大統領令を受けた政府機関の対応についても紹介がなされた。加えて、SBOM の普及促進に向けた複数の課題が提示された。具体的な課題として、SBOM は様々な分野において応用できる仕組みとする必要があること、クラウドや SaaS 等のサービスに対しても SBOM の検討が必要であること、SBOM の自動生成・自動配布の方法について検討がまだ不足していること、SBOM の信頼性確保の方法が十分に検討されていないこと等が意見された。Friedman 博士の発表の最後では、日本で作成されたツールやガイド等と連携する意向が意見された。

自由討議では、国内から参加した関係組織と Friedman 博士との議論を実施した。具体的には、国内関係組織から挙げられた質問に対して Friedman 博士が回答するという質疑応答形式での議論が行われた。まず、SBOM におけるソフトウェアの出所に対する保証方法に関する質問について、出所の確認は大統領令におけるベースラインの項目ではなく、ユーザーから求められる項目であり、今後はデータの整合性を確保することから始め、次に出所等の情報の取得方法を検討していく必要があるとの回答が得られた。VEX において活用するデータに関する質問について、各企業のこれまでの製品セキュリティへの投資や実施状況を加味するとの回答が得られた。SBOM による利便性が導入コストを上回った事例に関する質問について、自社が開発しているソフトウェアの SBOM を公表し、サプライチェーン全体での SBOM 生成を促した企業の事例が挙げられた。また、米国政府は中小企業にとって不利益にならないよう取組を進めているとの回答も得られた。SBOM ツールにおける偽陽性や偽陰性の課題に関する質問について、米国では SBOM を直接比較することで対応したという例が共有されたほか、ツール同士の協調にも取り組んでいるとの回答が得られた。

3. OSS の利活用等に関する事例の調査

令和2年度事業で作成した事例集と、2章で新たに調査した結果を踏まえ、「自社の製品・サービスにOSSを活用する企業」及び「OSS又はOSSを含む製品・サービスを利用する企業」に対し、OSSを利活用する際に行っている特筆すべき取組について、調査を行い、事例集を拡充した。3.1節に事例集の拡充に関する観点整理、3.2節に公開文献調査、3.3節にヒアリング調査、3.4節に具体的な事例集の拡充の結果整理を報告する。

3.1. 事例集の拡充に関する観点整理

事例集の拡充については、予め収集すべき情報や調査の観点を設定し、収集すべき情報や調査の観点に基づき、公開情報調査を実施した。さらに、公開情報調査によって収集した事例からヒアリング調査を実施し、詳細な情報を得た内容を踏まえ、事例集の拡充を行った。

令和2年度事業における調査、及び作成した事例集から、本調査で確認すべき調査項目を以下のように設定した。OSSの利活用等に関する事例を拡充するための本章での調査観点と3.2節の文献調査や3.3節のヒアリング調査における具体的な調査項目の関係を以下に示す。

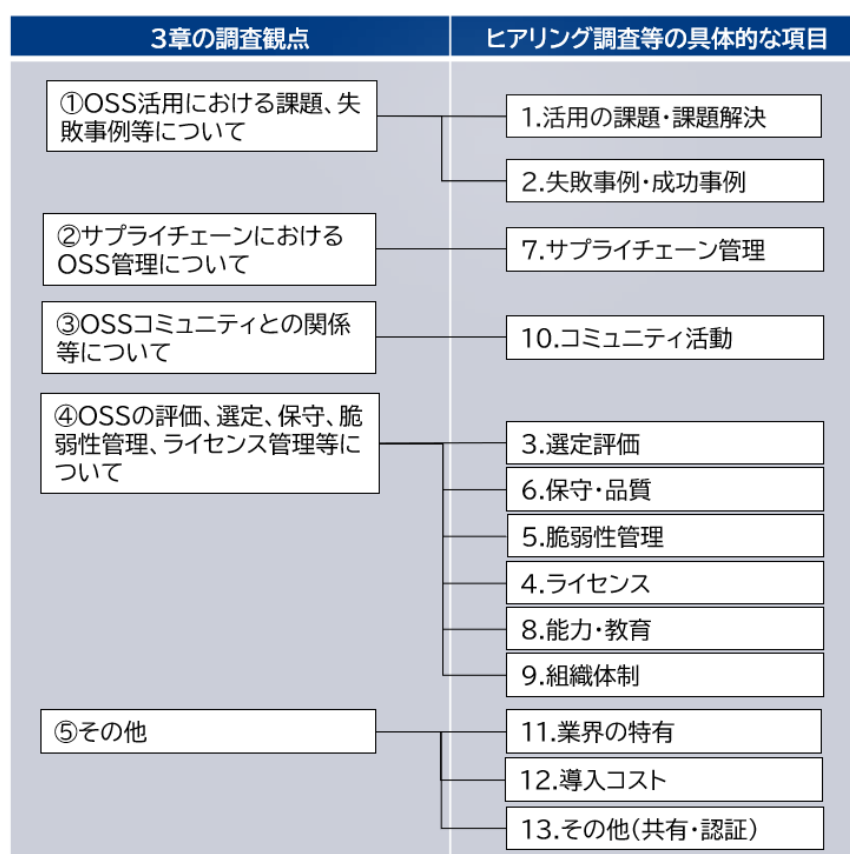


図 3-1 調査の観点とヒアリング調査等の項目との関係

3.2. 公開文献調査

事例収集のための公開文献調査の概要を表 3-1 に示す。表内の●は調査によって確認できた項目である。また、3.2.1 以降に各事例の課題及び課題解決として確認できた各項目の概要を報告する。

表 3-1 公開文献調査概要

想定調査対象	1.活用の課題・課題解決	2.失敗事例・成功事例	3.選定評価	4.ライセンス	5.脆弱性管理	6.保守・品質	7.サプライチェーン管理	8.能力・教育	9.組織体制	10.コミュニティ活動	11.業界の特有	12.導入コスト	13.その他(共有・認証)
1. オリンパス株式会社	●			●	●				●				
2. Zendesk, Inc.	●			●					●				
3. Verizon Communications Inc.	●			●									●
4. JS Foundation (現 OpenJS Foudation)	●			●									●
5. Applause App Quality, Inc.	●			●									●
6. APRESIA Systems 株式会社	●			●					●				
7. ウイングアーク 1st 株式会社	●					●		●	●				
8. 株式会社システムインテグレータ	●				●				●				●

3.2.1. 公開文献事例 1（オリンパス株式会社）

オリンパスでは、自社製品で利用している OSS の管理対応として検証ツールを導入。これによってサービスに内包されている OSS の把握が可能となり、人力ではカバーできなかった OSS をピックアップすることができるようになった。その結果、ライセンス違反の大幅な手減が可能となった。また、機能の一つとして搭載されている脆弱性の監視機能の活用により、脆弱性の早期発見が可能となった。公開文献事例 1 に関する詳細について確認できた概要を以下に報告する。

表 3-2 公開文献事例 1 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業
企業名	オリンパス株式会社
収集した URL	https://www.aperza.com/catalog/page/7533/20184/
1.活用の課題・課題解決	製品・サービスに内包している OSS の把握及びライセンス違反のリスクを大幅に低減することに成功。また、自動化ツールによる OSS ガバナンスを効率化させ、ポリシーの定着を促進。
4.ライセンス	検証ツールに高性能な OSS データベースを備えており、各 OSS に関して、適用されるライセンスや開発コミュニティの活動状況などの豊富かつ重要な情報を参照が可能。
5.脆弱性管理	活用している OSS に関連する脆弱性の情報をマッチングしタイムリーなセキュリティ対策を可能にするとともに、脆弱性の監視機能を備えており、活用している OSS に新たな脆弱性が検出された場合に通知できる。
9.組織体制	知識共有というアプローチによって開発者が確実に使用ポリシーを理解させることができた。

3.2.2. 公開文献事例 2（Zendesk, Inc.）

Zendesk は、開発チームの増加に伴う OSS 利用数の増加を課題としていた。以前から OSS 管理ツールを使用していたが、開発チームの OSS 管理工数及び法務チームとの情報連携の効率化を目的とし、FOSSA を導入した。これによってソフトウェア開発ライフサイクル全体での OSS 管理とリアルタイムでライセンスの確認が行われるようになり、OSS 管理工数が削減された。また、ツールを介して法務チームと連携を取ることで、ワークフローが効率化された。公開文献事例 2 に関する詳細について確認できた概要を以下に報告する。

表 3-3 公開文献事例 2 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業
企業名	Zendesk, Inc.

項目	概要
収集した URL	https://www.hitachi-solutions.co.jp/-/media/Project/DefaultSite/Products/pdf/f/FOSSa_case01_zendesk.ashx
1.活用の課題・課題解決	開発チームの数が増えることで OSS の利用が増加し、従来の OSS 管理ツールでは対応が追いつかない状況となってきた。また、Zendesk はオープンソース・ライセンスに準拠するための包括的でリアルタイムのアプローチも必要としていたため、管理ツール FOSSA の導入を検討した。FOSSA は CI/CD パイプラインに統合し、ソフトウェア開発ライフサイクル全体でライセンス問題と修復ガイダンスの洞察を提供するように設計されているので、Zendesk が求める継続的コンプライアンスに最適なソリューションとなった。
4.ライセンス	FOSSA は GitHub のレポジトリとビルドサーバーの両方でコードの依存関係をスキャンすることで、デプロイ前にすべてのオープンソースコンポーネントと関連するライセンスをカタログ化し、シフトレフト及びコンプライアンスワークフローの自動化を実現。
9.組織体制	法務チームとエンジニアリングチームのワークフローを効率化

3.2.3. 公開文献事例 3（Verizon Communications Inc.）

Verizon は多層構造となっているライブラリの管理維持が困難であるという課題を持っていた。FOSSA を導入することで各ライブラリに含まれる OSS の把握及びライセンス管理が可能となった。公開文献事例 3 に関する詳細について確認できた概要を以下に報告する。

表 3-4 公開文献事例 3 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業
企業名	Verizon Communications Inc.
収集した URL	https://www.hitachi-solutions.co.jp/-/media/Project/DefaultSite/Products/pdf/f/FOSSa_case02_verizon.ashx
1.活用の課題・課題解決	モバイルアプリは多くのライブラリで構成され、またそのライブラリはさらに多くのライブラリを前提として動いているので、オープンソース・ライセンスのコンプライアンスを維持するのは困難になってきた。FOSSA を導入し、CI/CD プロセスに統合することで各コードに含まれるすべての OSS を特定することができた。
4.ライセンス	FOSSA を活用してオープンソース・ライセンスのコンプライアンス遵守を自動化することで、法務チームとエンジニアリングチームの両方で膨大な工数を削減できた。
13.その他（共有・認証）	FOSSA を活用し、開発者及び DevOps チームがコンプライアンスに準拠したコードを大規模に展開することを可能にした。

3.2.4. 公開文献事例 4（JS Foundation）

JS Foundation（現 OpenJS Foundation）はライセンス管理を課題としており、解決のため FOSSA の導入を実施した。ツールを導入したことで、プロジェクト内のすべてのソースコードがスキャンできるようになり、ライセンス違反が自動で検出されるようになった。公開文献事例 4 に関する詳細について確認できた概要を以下に報告する。

表 3-5 公開文献事例 4 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業
企業名	JS Foundation（現 OpenJS Foundation）
収集した URL	https://www.hitachi-solutions.co.jp/-/media/Project/DefaultSite/Products/pdf/f/FOSSa_case03_js-foundation.ashx
1.活用の課題・課題解決	各プロジェクトの自立性を維持しながら、主要プロジェクトのライセンス・コンプライアンスと依存関係の追跡を監視、管理、維持するための、信頼性の高い自動化された方法の確立が課題となっていた。FOSSA はプロジェクト内のすべてのソースファイルと依存関係をスキャンしてライセンス違反を検出することでコンプライアンス遵守の自動化及び省力化を実現できた。
4.ライセンス	自動化されたライセンス・コンプライアンスの継続的なスキャンにより、法務チームとエンジニアリングチームの両方で手作業による調査の負担が軽減された。
13.その他（共有・認証）	自動化されたライセンス・コンプライアンスのスキャン結果を、法務チームとエンジニアリングチームに展開することで両チームの情報展開及び情報統一が可能となり、調査に係る負担が軽減された。

3.2.5. 公開文献事例 5（Applause App Quality, Inc.）

Applause App Quality は FOSSA 導入により、「オープンライセンスに準拠しているか」という顧客からの問い合わせに迅速に対応することに成功した。また、FOSSA に搭載されている様々なレポート機能を対顧客のみではなく、社内の開発チームとの情報連携にも活用している。公開文献事例 5 に関する詳細について確認できた概要を以下に報告する。

表 3-6 公開文献事例 5 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業
企業名	Applause App Quality, Inc.
収集した URL	https://www.hitachi-solutions.co.jp/-/media/Project/DefaultSite/Products/pdf/f/FOSSa_case04_applause.ashx

項目	概要
1.活用の課題・課題解決	オープンソースのライセンスを特定して文書化することにおいて、その情報を常に最新かつ正確に保つことが大きな課題となっていた。FOSSA を利用することでレポジトリの統合、チームメンバー間のシームレスなコミュニケーション、コンポーネント変更に応じたレポートの更新などを簡単に実施できるようになった。
4.ライセンス	FOSSA を使うことで包括的なオープンソースのリストを作成するための手動プロセスが不要になった。
13.その他（共有・認証）	顧客やパートナーが、APPLAUSE がオープンライセンスに準拠していることを求めており、これを証明するための情報をエクスポートしレポートとして活用することができるようになった。

3.2.6. 公開文献事例 6（APRESIA Systems 株式会社）

APRESIA Systems では、OSS の管理が各部署やプロジェクトによってバラバラであった。リスト化している部署はあっても、ライセンスまでは管理できてなかった。Black Duck Protex を導入することで、利用しているソースコードに含まれるライセンス情報を把握、管理できるようになった。公開文献事例 6 に関する詳細について確認できた概要を以下に報告する。

表 3-7 公開文献事例 6 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業
企業名	APRESIA Systems 株式会社
収集した URL	https://www.hitachi-solutions.co.jp/oms/case02/
1.活用の課題・課題解決	多種多様な OSS を使用していたが、管理は統一されたものではなく、訴訟などのリスクから防衛し、コンプライアンスを強化したいという要望が挙げられていた。 「オープンソース管理ソリューション」を導入し、イーサネットスイッチ製品の OS の開発に適用した。管理を一元化できたことで、ライセンス違反等のリスクを低減することができた。
4.ライセンス	膨大なソースコードのライセンスチェックを実現。 1. オープンソースの意図せぬ混入を防止。 2. 人の判断ではほぼ不可能な OSS の類似コードを検出。
9.組織体制	OSS のライセンス訴訟などのリスクから防衛し、コンプライアンスを強化

3.2.7. 公開文献事例 7（ウイングアーク 1st 株式会社）

ウイングアーク 1st では、OSS の管理を品質保証プロセスの中に、いかに組み込むかが課題の一つとなっていた。OSS 管理ツールとして Black Duck Protex を使用していたが、効果的には利用できておらず、品質保証の中のリスク管理として利用できていないという点に

おいても、課題があった。

様々な課題を解決するべく、「オープンソース管理ソリューション」を導入し、OSS に関する社内教育、運用時におけるツールの活用方法の提案等を受け、リスク管理、工数削減、脆弱性管理など、様々な導入効果を得ることができた。公開文献事例 7 に関する詳細について確認できた概要を以下に報告する。

表 3-8 公開文献事例 7 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業
企業名	ウイングアーク 1st 株式会社
収集した URL	https://www.hitachi-solutions.co.jp/oms/case01/
1.活用の課題・課題解決	・ OSS の管理を品質保証プロセスの中に、いかに組み込むかが課題となっていた。また OSS に関する問い合わせ・調査などに、非常に時間がかかることも課題として挙げられていた。 この課題については、知識・ノウハウが豊富な日立ソリューションズによりソリューションを提供、品質保証プロセスに組み込むことができた。 ・ OSS の相談・調査などに、非常に時間がかかる 専任エンジニアによる「よろず相談窓口」で、迅速な対応が実現可能になった。
6.保守・品質	相談内容の広がりや量の増大に伴い、ウイングアーク 1st のオフィス内に、知識・ノウハウが豊富な専任エンジニアが常駐。より迅速に応えられるコンサルティングサービスの提供が可能となった。
8.能力・教育	扱い方、権利関係など OSS への理解を深めるため、開発メンバー向け教育の実施を決定。日立ソリューションズとともに内容や方法等を検討、実施した。
9.組織体制	開発メンバーに対して向け教育の実施を決定。日立ソリューションズとともに内容や方法等を検討

3.2.8. 公開文献事例 8（株式会社システムインテグレータ）

システムインテグレータでは、OSS の脆弱性管理において、担当者が定期的に脆弱性情報サイトを見に行き、情報を確認していたため、確認の工数が多くかかってしまい、また担当者によって参照するサイトが異なる等やり方もばらばらであり、情報のばらつきや検知スピードの担保が困難であった。管理ツールを導入したことにより、脆弱性管理にかかる工数自体が減り、チェックの抜け漏れ防止の精度が向上した。公開文献事例 8 に関する詳細について確認できた概要を以下に報告する。

表 3-9 公開文献事例 8 概要

項目	概要
分類	自社の製品・サービスに OSS を活用する企業

項目	概要
企業名	株式会社システムインテグレータ
収 集 し た URL	https://yamory.io/blog/case-system-integrator/
1.活用の課題・課題解決	担当者ベースで調査を実施していたため、チェック漏れや作業工数の増加といったリスクが発生する懸念があった。YAMORY 利用による作業の自動化を実現し、セキュリティにかかる工数削減（1 チームあたり 2～3 人日）の実現でき、脆弱性のチェックがより高品質で標準化させることができた、
5.脆弱性管理	脆弱性への対応優先度が自動で分類されることで、脆弱性管理にかかる工数自体が減った。
13. その他（共有・認証）	脆弱性が多いライブラリに関しては他のライブラリの使用を検討する等、ライブラリ自体の選定にも活用できた。

3.3. ヒアリング調査

事例収集のためのヒアリング調査の概要を表 3-10 に示す。表内の●は調査によって確認できた項目である。
なお、下記項目の No. 4 (A 社)、No. 5 (B 社) 以外の具体的な内容及び詳細は事例集を参照。

表 3-10 ヒアリング調査概要

想定調査対象	1.活用 の課 題・課 題解決	2.失敗 事例・ 成功事 例	3.選定 評価	4.ライ センス	5.脆弱 性管理	6.保 守・品 質	7.サブ ライチ ェーン 管理	8.能 力・教 育	9.組織 体制	10.コ ミュニ ティ活 動	11.業 界の特 有	12.導 入コス ト	13.そ の他 (共 有・認 証)
1. OSSTech 株式会社	●			●	●	●			●	●			
2. 三菱電機インフォメーションシステムズ株式 会社			●	●	●	●	●		●	●			
3. SCSK 株式会社	●		●	●	●	●	●	●	●				
4. A 社	●							●		●			
5. B 社								●		●			
6. 日本電気株式会社					●	●	●	●				●	

想定調査対象	1.活用 の課 題・課 題解決	2.失敗 事例・ 成功事 例	3.選定 評価	4.ライ センス	5.脆弱 性管理	6.保 守・品 質	7.サブ ライチ ェーン 管理	8.能 力・教 育	9.組織 体制	10.コ ミュニ ティ活 動	11.業 界の特 有	12.導 入コス ト	13.そ の他 (共 有・認 証)
7. ヤフー株式会社	●		●			●	●		●			●	
8. 株式会社ラキール			●	●	●				●			●	

3.4. 事例集の拡充の結果整理

前述の公開文献調査及びヒアリング調査の結果、主にヒアリング調査を実施した 5 件を事例として新規に追加した。また、既存の NEC の事例について PSIRT 及び脆弱性の管理の観点で追加した。新規に追加した 5 件については、以下のような事例である。なお、詳細について別紙の OSS 活用事例集拡充版を参照。

- OSSTech 株式会社：構成管理では、ビルドシステム等の OSS に備わっている機能を活用しながら、ソフトウェアの依存関係を管理している事例。
- 三菱電機インフォメーションシステムズ株式会社：OSS を長期間利用するうえでの脆弱性管理やアップデート対応に係るコストの考え方を顧客と事前に合意する等の事例。
- SCSK 株式会社：開発プロジェクトにおけるリスクの第三者チェック機能である PMO 組織の設立や各事業グループに OSS 管理担当者を設置することで組織体制を構築している事例。
- ヤフー株式会社：ヤフー全体で OSS の取り扱いに関するガバナンスが取れており、複数の審査を経たうえで OSS の利用が可能となっている事例。
- 株式会社ラキール：OSS の利用を全社的に促進しており、新しい OSS を利用する際には、開発部と OSS 管理部門のリーダー同士がリスクの確認や懸念点を議論する会議で決定する事例。

4. ソフトウェアの利活用に係るセキュリティリスク、課題及び対応策の検討

前述の2章、3章の調査結果も踏まえ、自社製品・サービスの一部としてソフトウェアを利活用する際に生じるセキュリティリスクや課題、セキュリティリスクに対応するための取組について、「製品・サービスにOSSを活用する企業」や「OSSを活用した製品・サービスを利用する企業」を対象に調査した。この結果を4.1節に報告する。また、調査した結果を踏まえて、セキュリティリスクへの有効性の高い対応策を検討した結果を4.2節に報告する。

4.1. 動向等に関する文献調査

3章の調査観点と公開文献調査やヒアリング調査の項目と、本章の調査観点（ソフトウェアの利活用に係るセキュリティリスク・課題及び対応策）の関係を以下に示す。

以降に調査した結果として、OSS活用における課題及び課題解決の参考事例を4.1.1節及び4.1.2節に、サプライチェーンにおけるOSS管理を4.1.3節に、脆弱性情報等の共有及び共有体制を4.1.4節に、OSSコミュニティにおける連携・脆弱性流通の現状を4.1.5節に報告する。



図 4-1 本章の調査観点とヒアリング調査等の項目との関係

なお、OSSの評価や選定、保守、脆弱性管理やライセンス管理（上図の3章の調査観点④）及びソフトウェア技術・認証（上図の4章の調査観点ソフトウェア技術・認証）の詳細については、その具体的な内容をヒアリング調査等で確認する必要があるため、本書の3.3節を参照。また、脆弱性情報の取扱いや共有などの参考となる独立行政法人情報処理推進機構の取組等については4.1.4.2を参照。

4.1.1. OSS 活用における課題、失敗事例等

OSS 活用の課題のひとつに、脆弱性情報の公開後の対応が遅れることによる被害の発生

及び被害の拡大が考えられる。以下に、OSS 活用における課題や失敗事例として近年の脆弱性の事例を以下に報告する。

4.1.1.1. OpenSSL Heartbleed 事例

ここでは OSS として提供されてシステムに組み込まれているライブラリあるいはサーバソフトウェアについての課題及び失敗事例について説明する。

4.1.1.1.1. OpenSSL の概略

OpenSSL は OSS の暗号通信ライブラリである。1995 年当時オーストラリアのプログラマ 2 名によって書かれ公開されていた SSL のオープンソース実装⁸⁷である SSLeay があった。1998 年に SSLeay の中心的な開発者が離れるタイミングで分岐したのが OpenSSL プロジェクトである。1998 年 12 月にバージョン 0.9.1 公開から OpenSSL としてバージョンアップを重ね、2021 年 9 月現在バージョン 3.0.0 が公開されている⁸⁸。

現在、OpenSSL は各種 RFC 規約のリファレンス実装というポジションとして扱われており、Web サービスを提供する httpd サーバーや PC やスマートフォン上の Web ブラウザだけに留まらず、VPN サーバー/クライアントなどにも暗号化通信ライブラリとして利用されている。

OpenSSL そのものはオープンソースであるが、OpenSSL ライブラリとしてベンダーから提供されるものの中には Federal Information Processing Standard (FIPS) 140-2, Security Requirements for Cryptographic Modules の認証プロセスである The Cryptographic Module Validation Program (CMVP)⁸⁹認証を取得しているものもある。GNU/Linux ディストリビューションのベンダーで認証をうけているモジュールを一部抜粋する。

- Red Hat Enterprise Linux 7 OpenSSL Cryptographic Module
- Oracle Linux OpenSSL Cryptographic Module
- SUSE Linux Enterprise OpenSSL Cryptographic Module
- Ubuntu OpenSSL Cryptographic Module

2021 年中に OpenSSL 関連で CMVP 認証 (FIPS140-2) を取得した製品は 30 あり、現在の CMVP 認証が有効となっている製品 (暗号モジュール) はトータルで 53 存在する⁹⁰。

4.1.1.1.2. OpenSSL Heartbleed の概要

OpenSSL Heartbleed は 2014 年 4 月に発見された暗号通信ライブラリ OpenSSL の脆弱性 (CVE-2014-0160) である⁹¹。2011 年 12 月に加わった接続継続性を確認するパケットのための拡張 (Heartbeat 拡張) のコードに脆弱性が含まれていた。特定の加工を施した攻撃パケ

⁸⁷ BSD ライセンスライクな独自ライセンスを採用している。また、当時は、まだ「オープンソース」という言葉はつかわれておらず、フリーソフトウェアの一種と呼ばれていた。

⁸⁸ TLS/SSL and crypto library <https://github.com/openssl/openssl>

⁸⁹ Cryptographic Module Validation Program (CMVP) <https://www.nist.gov/programs-projects/cryptographic-module-validation-program-cmvp>

⁹⁰ Cryptographic Module Validation Program (CMVP) <https://www.nist.gov/programs-projects/cryptographic-module-validation-program-cmvp>

⁹¹ CVE-2014-0160 <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160>

ットを Heartbleed の脆弱性を持つサーバーに送ると、そのサーバーが保持しているメモリ内容がリークして戻ってくる。そのためサーバーのメモリ内に保持されている秘密鍵やあるいはパスワードなど機密情報が漏れ出す危険性があった。その約 3 年後となる 2017 年 1 月の段階においてもインターネット上の 180,000 サイト近い Web サーバーが、OpenSSL Heartbleed の脆弱性を持ったまま稼働していた⁹²。

4.1.1.1.3. OpenSSL Heartbleed 脆弱性の影響

OpenSSL Heartbleed はクライアント側からサーバー側に対してメモリの内容をリークさせるが、どのような情報がリークするのかは、サーバー側のメモリの状態に依存する。システムに致命的な情報がリークするかは確率的ではあり評価は難しいが、一方で、情報を守るべきツールが秘匿情報をリークするという点においては危険な脆弱性である。

インターネット上での主力 Web サーバーに Apache と nginx があり、両者とも標準的な暗号ライブラリとして OpenSSL を利用している。2014 年当時のインターネット上でのシェアをみると Apache が約 38%、nginx が約 15%となっていた⁹³。つまり潜在的にはインターネット上の Web サーバーの約 53%が OpenSSL Heartbleed の脆弱性が存在しているということになり、広範囲に影響が及ぶ可能性があった。ただし、OpenSSL HEARTBEATS そのものは、コンパイル時のオプションで ON/OFF が可能である(-DOPENSSL_NO_HEARTBEATS)。OpenSSL のディストリビューションに用意されているパッケージでは、HEARTBEAT 機能は含まれているが、ソースコードからビルドして利用しているユーザーの場合、必ずしも HEARTBEAT 機能を含んでいるとは限らない。

4.1.1.1.4. 自由度の高さと脆弱性

商用版、コミュニティ版のいずれもディストリビューションの標準をそのまま使っているならば、ディストリビューションの脆弱性対応アップデートが作成され、システムアップデートの時点でライブラリは入れ替えられる。しかし、何らかの理由⁹⁴でソースコードを手し、自前でビルドをしている場合がある。そのようなシステムでは、ソースコードのアップデートを自ら把握してメンテナンスしていく必要がある。現在稼働している Web サーバーが OpenSSL Heartbleed の脆弱性を含んでいるかどうかは、脆弱性を含む OpenSSL のバージョンであることの確認以外に、拡張オプションとして指定されているかどうかも確認しなければならない。また、入れ替えるためには、ビルド環境を再構築し、ビルド作業を行わなければならない。このようにユーザーが自由度の高い独自ビルドの形で導入しようとする場合、緊急に行わなければならないような脆弱性対応アップデートなどのメンテナンスのコストも考慮に入れておく必要がある。

⁹² Heartbleed Lingers: Nearly 180,000 Servers Still Vulnerable <https://www.bankinfosecurity.com/blogs/nonstop-heartbleed-nearly-200k-servers-still-vulnerable-p-2381>

⁹³ February 2014 Web Server Survey <https://news.netcraft.com/archives/2014/02/03/february-2014-web-server-survey.html>

⁹⁴ ハードウェアアーキテクチャーに依存したチューンナップオプション利用などが考えられる。

4.1.1.2. ShellShock (bash の脆弱性) 事例

4.1.1.2.1. UNIX 系システムにおけるシェルの位置づけ

bash は UNIX 系システムのシェルプログラムの 1 つで、多くの UNIX 系システムで標準的に用意されている。シェルプログラムとは、コマンドインタプリターのプログラムで UNIX ではユーザーが端末でコマンドを動作させる時のインタフェースプログラムとして使われている。また、シェルで書かれたスクリプトは UNIX 系システムにおいては色々な場面で実行プログラムとして利用されている。GNU/Linux のディストリビューションでは標準的に bash をシステムで利用しているだけではなく、各種 UNIX 系システム上で同一のシェルプログラムを動作させるためのプラットフォームとして bash が使われるケースもある。故に、bash は、システムの標準で用意しているコマンドが bash で記述されている、他のアプリケーションのサブシステムとして内部で呼び出されるスクリプトとして使われている、あるいはユーザーが処理を記述するためのスクリプトとして用意している等、あらゆる場面で利用されている。このようなシステムからユーザーまで幅広く使われているコマンドインタプリターに脆弱性が発生した場合、大きな影響を与える。

4.1.1.2.2. bash の脆弱性 (ShellShock) 概要

ShellShock とは、2014 年 9 月 24 日に公開された bash の脆弱性 (CVE-2014-6271) である。環境変数に名前のない関数とシェルコマンドを設定したうえで bash を実行すると、環境変数に設定したシェルコマンドが実行されるため、脆弱性のある bash が実行されるタイミングで外部から環境変数を与えることが可能な場合、任意のコマンドが実行できる。脆弱性の危険度を示す CVSS v2 の値は最大値の 10.0 であった。最悪の場合、管理者権限 root によって任意のコードが実行される可能性がある。以下に攻撃方法の例をいくつか挙げる。

- **Web アプリを經由しての Web サーバーの乗っ取り**

Web アプリが内部で bash を利用している場合、呼び出し側のユーザーエージェントに攻撃コードを組み込むことが可能である。任意のコードが実行可能なので、攻撃コードに RAT (リモート・アクセス・ツール) を入れれば、外部からサーバーにリモートシェルを立ち上げることが可能である。また Web アプリ連動するデータベースの内容すべてを外部に流出させることも可能である。

- **悪意のある DHCP を用意して接続した PC の乗っ取り**

DHCP は PC などを LAN に接続する際に動的な IP アドレスを割り振るための仕組みである。PC を無線 LAN や有線 LAN に接続した際、IP アドレス、DNS サーバー、ルーティングなど、ネットワーク接続に必要な情報を DHCP サーバーから DHCP クライアントに送るので PC がネットワーク環境を設定できる。その時に、DHCP サーバー側から DHCP クライアント側に環境変数を送ることが可能である。もし、DHCP サーバーから攻撃コードを送られ、かつ DHCP クライアント側でコンフィギュレーションスクリプトが bash で書かれていた場合、任意のコードが実行されることになる。この場合、DHCP クライアントは root 権限で動作しているため、攻撃が成功した場合

合、任意の実行コードが `root` 権限で動作することになる。最悪なケースを想定すると、この問題を抱えている PC が悪意の攻撃が仕掛けられているオープン Wi-Fi に自動接続した瞬間にシステムは乗っ取られることが考えられる。

- **ファイル転送 `sftp` を使ったサーバーの乗っ取り**

ファイル転送 `sftp` は SSH サーバーのサブシステム `sftp-server` を利用している。ネットワーク上のユーザーがファイル転送 `sftp` を使い、ファイルサーバーにファイルをアップロード・ダウンロードすることが可能である。SSH にはクライアント側からサーバー側に環境変数を送るという機能 (`AcceptEnv/SendEnv`) がある。ファイル転送 `sftp` も SSH と同様にクライアント側からサーバー側に環境変数を送る機能を使える。これにより本来はファイル転送のみしか利用が許されていないユーザーが、攻撃コードを送り込むことで `sftp` の権限において任意のコマンドをファイルサーバー上で実行することが可能となる。

ここでは種類の違う 3 つのケースを取り上げたが、コマンドインタプリタである `bash` は色々な場所で使われている。GNU/Linux のシェルは `bash` ではなく `dash`⁹⁵といった別実装のシェルコマンドが使われているケースもあり、どの部分に `bash` が使われているのか探し出すには労力を要する。また、探してリストアップしたとしても、すべて探しきれているかは不透明である。利用するソフトウェアの提供元が `bash` のスクリプトを含んでいるか否かをマニフェスト等で示さない限り、エンドユーザー側で脆弱性の影響範囲を探し出すことは非常に困難である。

4.1.1.2.3. 潜在的な脆弱性が長期間存在

`bash` の脆弱性 (CVE-2014-6271) の特徴は、潜在的な脆弱性が長期間潜んでいた点である。この脆弱性の元となったコードは構文解析部分にあるが、この部分はリリース初期の 1989 年から脆弱性が発見されて修正される 2014 年まで約 25 年間修正されていない。ShellShock で使われる環境変数の攻撃コードは、通常では使わないような記述なので、テストケースといった方法では見つけることは難しいのではないかと考えられている⁹⁶。

CVE-2014-6271 が公表された後、世界各地の OSS デベロッパー有志により集中的にコードの監査が行われ、2014 年 9 月 24 日から 9 月 28 日までに合計 6 の脆弱性が公開された (公開時には修正されている)。最初の CVE が公開された 1999 年から 2021 年 11 月末までの約 22 年間に `bash` の CVE が公開されたのは大小含めて合計 17 であるが、うち 6 がこの数日に集中している。問題が発生した後に OSS コミュニティの集中的な対応により極めて短時間で状況が刻々と変化したケースといえる。

⁹⁵ `dash` は `bash` とは異なる実装であるため `bash` の CVE-2014-6271 に相当する脆弱性は `dash` には存在していない。

⁹⁶ すずきひろのふ「マジメだけどおもしろいセキュリティ講義 事故が起きる理由と現実的な対策を考える」技術評論社、2017 年

4.1.1.3. Apache Struts2 脆弱性事例

4.1.1.3.1. Apache Struts 2 の概略

Apache Struts は Apache Software Foundation の Apache Struts プロジェクトが開発しオープンソースとして公開している Java Web アプリケーションのフレームワークである。Apache Software Foundation の Apache Struts プロジェクトは 2000 年からスタートしており、オープンソース・プロジェクトとして 20 年以上の歴史を持つ。Struts1 は既に 2013 年 4 月にサポートを終了しており、現在は Struts2 が開発・提供されている。動的な Web サービスを提供するための Java Web アプリケーションのフレームワークなので EC サイトや銀行サイトなどにも広く使われている。そのため攻撃者の興味を引く対象であり、また実際に攻撃を実証するコードが複数公開されている⁹⁷。セキュリティのための緊急アップデートが公開されても、稼働中の Web アプリケーションとの不整合が発生しないかといった動作確認などが必要となるため単純な置き換え作業だけに留まらず移行作業計画などが必要となる場合が多いと考えられる。

4.1.1.3.2. Apache Struts 2 の脆弱性 (CVE-2017-5638) 概要

Apache Struts 2 の脆弱性 CVE-2017-5638/S2-45^{98,99,100} (以降 CVE-2017-5638 と呼ぶ) は攻撃者が加工した HTTP リクエストを Struts アプリケーションサーバー (Apache Struts 2 を使用するアプリケーションサーバー) へ送ることで、任意のコードを実行することが可能となる脆弱性である。Apache Software Foundation は 2017 年 3 月 2 日に CVE-2017-5638 対応のバージョンを公開したが、その対応バージョンが広く適用される前に攻撃が行われ不正アクセスの被害が発生した。日本国内でも 2017 年 3 月 9 日には GMO ペイメントゲートウェイに対して不正アクセスが発生する¹⁰¹など、脆弱性対応のための対応時間が短いことが特徴的な事例であるといえる。一方で、政府統計の総合窓口サイト e-Stat が提供する「地図による小地域分析 (jSTAT MAP)」も同じ脆弱性を攻撃された。前述のとおり脆弱性が公開された時点の 2017 年 3 月に対策が実施されていなく、2017 年 4 月 11 日に不正アクセスが認知されるなど¹⁰²、脆弱性対応や不正アクセスの認知の遅れが目立っている事例となっている。

4.1.1.3.3. CVE-2017-5638 の影響

CVE-2017-5638 の特徴的な点は、脆弱性対応のバージョンが公開されてから数日で攻撃コードが公開されたことである。最初の脆弱性情報公開時/脆弱性対応バージョンの公開時に、攻撃容易性に関しての言及はなかった。そのため利用者は緊急的な対応は取っていなかつ

⁹⁷ Apache Struts2 の脆弱性対策情報一覧 https://www.ipa.go.jp/security/announce/struts2_list.html

⁹⁸ CVE-2017-5638 <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-5638>

⁹⁹ CVE-2017-5638 Detail <https://nvd.nist.gov/vuln/detail/cve-2017-5638>

¹⁰⁰ S2-045 <https://cwiki.apache.org/confluence/display/WW/S2-045>

¹⁰¹ 不正アクセスに関するご報告と情報流出のお詫び https://www.gmo-pg.com/corp/newsroom/pdf/170310_gmo_pg_ir_kaiji.pdf

¹⁰² 地図による小地域分析 (jSTAT MAP) における不正アクセス https://www.soumu.go.jp/menu_news/s-news/01toukei09_01000023.html

た。そこから数日で CVE-2017-5638 を使った効果的な攻撃方法が開発され、実際の攻撃が発生し、脆弱性対応が終わっていないサイトで被害が発生した。

脆弱性情報 CVE-2017-5638 の公開から一連の攻撃に至るまでの時系列を以下に示す。短い期間で状況が急激に悪化していったのを見て取れる。なお、時系列の日付はドキュメントにあるタイムスタンプを参考にしているため、各情報が発信された場所のタイムゾーンに依存して実際には前後している可能性があることに注意されたい。

- 2017 年 3 月 2 日：Apache Software Foundation が CVE-2017-5638 対応のバージョンを公開し S2-45 としてサイト上に告知する。
- 2017 年 3 月 6 日：同サイト上の告知内容に悪意のあるコードによる攻撃が可能であると情報が追記される。
- 2017 年 3 月 7 日：Exploit Database サイトに CVE-2017-563 を使ったリモート実行の実証コードが公開¹⁰³される。
- 2017 年 3 月 8 日：NTT セキュリティ・ジャパン Twitter 公式アカウントから Apache Struts 2 の脆弱性 S2-045（CVE-2017-5638）に対する攻撃が発生していることを警告¹⁰⁴するツイートが発信される。
- 2017 年 3 月 9 日：JPCERT/CC から JPCERT-AT-2017-0009「Apache Struts 2 の脆弱性（S2-045）に関する注意喚起」¹⁰⁵が公開される。JPCERT/CC においても任意のコードが実行できることを確認したと報告。また脆弱性対応の方法を提示。同時に脆弱性対策情報ポータルサイト JVN において JPCERT/CC 及び IPA が「Apache Struts2 に任意のコードが実行可能な脆弱性緊急」を告知する¹⁰⁶。
- 2017 年 3 月 9 日：同日深夜 GMO ペイメントゲートウェイが JPCERT/CC 及び IPA の注意喚起が公開されたタイミングで自社のシステムをチェックした所、外部からの攻撃があり、攻撃が成功した可能性が高いことを確認¹⁰⁷。翌日未明にかけて対応。

4.1.1.3.4. 緊急対応と時間的制約の問題点

Apache Struts2 脆弱性事例では緊急メンテナンスが可能であったか否かが被害の明暗を分けたと考えられる。本事例では脆弱性公表から数日で現実的な悪意のある攻撃が大規模に行われている。これはゼロデイ攻撃に近く、脆弱性の公表があった後に通常のメンテナンスのローテーションにあわせて後日スケジュールを計画し、実施するという時間的余裕はなかったといえる。このように、脆弱性が公表されてからすぐに攻撃が発生し、早急に対処しなければ被害が及ぶような状況を考えると情報収集も含め事前準備がなければ対応は難しいと考えられる。

¹⁰³ Apache Struts 2.3.5 < 2.3.31 / 2.5 < 2.5.10 - Remote Code Execution <https://www.exploit-db.com/exploits/41570>

¹⁰⁴ NTT セキュリティ・ジャパン株式会社 公式アカウント
https://twitter.com/GlobalNTT_JP/status/839132398210031616

¹⁰⁵ Apache Struts 2 の脆弱性 (S2-045) に関する注意喚起 <https://www.jpcert.or.jp/at/2017/at170009.html>

¹⁰⁶ JNVN#93610402 Apache Struts2 に任意のコードが実行可能な脆弱性 緊急
<https://jvn.jp/vu/JNVN#93610402/>

¹⁰⁷ 不正アクセスに関するご報告と情報流出のお詫び https://www.gmo-pg.com/corp/newsroom/pdf/170310_gmo_pg_ir_kaiji.pdf

4.1.2. OSS 活用における課題解決、成功事例等

OSS 活用における課題解決のひとつにセキュリティ設定共通化手順がある。以下に OpenSCAP の取組みを報告する。

4.1.2.1. Linux ディストリビューションにおける OpenSCAP の活用

4.1.2.1.1. SCAP

SCAP (Security Content Automation Protocol) は NIST が開発した情報セキュリティ対策の自動化と標準化を目的としたセキュリティ設定共通化手順である^{108, 109}。米国政府機関及び米国国防総省の情報システムのセキュリティを強化することを義務づける法律である連邦情報セキュリティマネジメント法 (FISMA 法/Federal Information Security Management Act) が 2002 年施行された。その法律の下で情報セキュリティ対策の一環としてシステムの構成を評価し、脆弱なバージョンのソフトウェアや脆弱な設定を検出するための自動化したコンテンツと、その交換するための仕様として SCAP が作成された。FISMA 法により米国政府機関は SCAP による実装を必要としており、それにより米国政府に情報システムを納める業者は SCAP のコンテンツ並びに SCAP ツールを用意することを求められることとなった。SCAP の仕様は NIST SP 800-126 rev 3¹¹⁰として NIST の Web サイトで公開されている¹¹¹。

SCAP は、セキュリティポリシー記述、テクニカルに脆弱性を検出するメカニズムやアセスメント評価の結果を提供するために、XCCDF (Extensible Configuration Checklist Description Format)、OVAL (Open Vulnerability and Assessment Language)、OCIL (Open Checklist Interactive Language) を用意している。SCAP レポートは収集された情報を標準化された形式で出力する。形式として ARF (Asset Reporting Format) と Asset Identification が用いられる。ARF は XML 形式の出力で、ARF 1.1 Schema (XSD1.0) となっている。Asset Identification は XML 形式の出力で、Specification for Asset Identification 1.1 となっている。

SCAP で利用される脆弱性情報は CVE がベースとなっている。整備された脆弱性情報流通及び管理の枠組みが用意されており、その情報を使ってシステム向けの XCCDF などが作られ、SCAP 仕様のツールを使い自動化されたシステム監査が行われるという流れになる。

4.1.2.1.2. OpenSCAP

OpenSCAP は SCAP のオープンソース実装である¹¹²。OpenSCAP サイトにおいてオープンソースであることの意義を次のように示している。

¹⁰⁸ Security Content Automation Protocol SCAP <https://csrc.nist.gov/projects/security-content-automation-protocol>

¹⁰⁹ SCAP(Security Content Automation Protocol)～情報セキュリティ対策の自動化と標準化を実現する技術仕様～ <https://www.ipa.go.jp/security/vuln/SCAP.html>

¹¹⁰ 2021 年 12 月現在

¹¹¹ SP 800-126 Rev. 3. The Technical Specification for the Security Content Automation Protocol (SCAP): SCAP Version 1.3 <https://csrc.nist.gov/publications/detail/sp/800-126/rev-3/final>

¹¹² OpenSCAP 公式サイト <https://www.open-scap.org/>

表 4-1 OpenSCAP サイト掲載のオープンソースの意義

原文	和訳
We believe that security is best done in the open. Open source development results in more scrutiny and allows community members to contribute without being held back by red tape, patents and secrets.	セキュリティはオープンで行うのが最善だと私たちは信じています。オープンソースでの開発はより精査され、かつコミュニティのメンバーが官僚主義、特許、秘密にとらわれることなく貢献できるようになります。

いくつかの主要なディストリビューション/コミュニティが参加している（表 4-2）。事務局にあたる機能を Red Hat が担当している。また OpenSCAP は SCAPVP（Security Content Automation Protocol Validation Program）を満たす SCAP ツールとして NIST に認証を受けている（図 4-2）。OpenSCAP は主要な GNU/Linux ディストリビューションで利用可能である。また Oracle Solaris のような非オープンソースのオペレーティングシステムでも提供されている。また OpenSCAP 単独ではなく特定のツールやシステム管理ツールにも組み込まれて採用されている。例えば NSA が開発し OSS として公開している The System Integrity Management Platform では OpenSCAP が利用されている。

表 4-2 OpenSCAP の採用状況

OpenSCAP をパッケージとして用意している GNU/Linux ディストリビューション	● Red Hat Enterprise Linux (Red Hat, Inc.) ● Fedora ● OpenSUSE (SUSE LINUX GmbH) ● Oracle Enterprise Linux (Oracle Corporation) ● Ubuntu (Canonical) ● Linux Mint ● Debian ● CentOS ● Scientific Linux
GNU/Linux 以外で OpenSCAP を採用しているオペレーティングシステム	● Oracle Solaris (Oracle Corporation)
OpenSCAP を採用しているプロジェクト	● Red Hat Satellite ● Anaconda ● NSA SIMP ● govready ● Preupgrade Assistant ● ManageIQ ● OSSEC (計画中)

Security Content Automation Protocol Validation Program SCAPVP	
f t	
142 Red Hat SCAP 1.2 Product Validation Record	
Validation Number:	142
Vendor:	Red Hat®, Inc.
Product Name:	OpenSCAP
Product Major Version:	1
Product Version Tested:	1.2.13
Tested Platforms:	Red Hat Enterprise Linux 6, 32 bit Red Hat Enterprise Linux 6, 32 bit Red Hat Enterprise Linux 7, 64 bit
SCAP 1.2 Capabilities:	Authenticated Configuration Scanner Common Vulnerabilities and Exposures (CVE) Validated Product Vendor Provided SCAP Information
Dates Tested:	11/22/2016 - 2/7/2017
Report Submitted:	2/8/2017
DTR Version:	NISTIR 7511 Rev. 4
Validation Test Suite:	version 1-2.0.0.0 (December 2012 release includes R600-1.2.0.0.1, R1900-1.2.0.0.2, R3300-1.2.0.0.3 out of cycle updates) version 1-2.0.1.0 (August 2013 release) version 1-2.0.2.0 (March 2014 release) version 1-2.0.3.0 (April 2015 release)

出所) NICT Security Content Automation Protocol Validation Program | CSRC¹¹³

図 4-2 NIST 公式サイト上の OpenSCAP 実装の認証状況

GNU/Linux ディストリビューションに付属している XCCDF ファイルはディストリビューションの種類やバージョンごとに用意されており、XML 形式で記述されている内容には、個々の監査項目に必要なシェルスクリプトの記述など必要なものが含まれている。http/https プロトコルによってネットワーク経由で外部定義ファイルをロードする機能があるので、必要に応じて外部のファイルを参照できる。

OpenSCAP コマンド“oscap”に XCCDF ファイルを指定して実行すると、自動的に XCCDF 内に用意しているシェルスクリプトが実行される。監査結果は XML 形式や HTML 形式で出力することが可能である。

```
$ sudo oscap xccdf eval --profile xccdf_org.ssgproject.content_profile_pci-dss --fetch-remote-resources --report report.html ssg-centos8-ds-1.2.xml
```

XCCDF ファイルで取り入れられている脆弱性情報は共通脆弱性識別子 CVE ベースである。米国の脆弱性流通及び管理の枠組みである Collaborative Vulnerability Metadata Acceptance Process (CVMAP) Program とリンクした形で利用される。その実装である OpenSCAP は米国政府に GNU/Linux ベースのシステムを納入する際に利用されるセキュリティポリシー監査ツールとして利用されている。また OpenSCAP を導入し定期的に自動監査することによって日々増加するシステムの脆弱性に対して厳密に対応できる可能性がある。また OpenSCAP は脆弱性をスキャンによって見つけるだけではなく、その脆弱性をどのように対応するか（修正するか）の情報も XCCDF ファイルには記載されており、その指示に従って修正することで脆弱性の問題を解決することができる。

¹¹³ <https://csrc.nist.gov/projects/scap-validation-program/validated-products-and-modules/142-red-hat-scap-1-2-product-validation-record>

4.1.2.2. OpenSCAP を応用して一般化した OSS 活用に結び付ける方法論

4.1.2.2.1. GNU/Linux における SCAP 利用

いくつかの GNU/Linux ディストリビューションではインストール時に SCAP の定義ファイルを使ってセキュリティポリシーを選択することが可能となっている。また監査用ツール `oscap` を利用してシステムのスキャンが可能となっている。以下に、open-scap.org が OSS 向け及び GNU/Linux ディストリビューション（商用も含む）向けに用意している SCAP Security Guide¹¹⁴のうち、主要なものをリストする。

表 4-3 SCAP Security Guide の主要リスト

ターゲット	SCAP Security Guide
Chromium	● Upstream STIG for Google Chromium
Debian 10	● Profile for ANSSI DAT-NT28 Average (Intermediate) Level ● Profile for ANSSI DAT-NT28 High (Enforced) Level ● Profile for ANSSI DAT-NT28 Minimal Level ● Profile for ANSSI DAT-NT28 Restrictive Level ● Standard System Security Profile for Debian 10
Fedora	● OSPP – Protection Profile for General Purpose Operating Systems ● PCI-DSS v3.2.1 Control Baseline for Fedora ● Standard System Security Profile for Fedora
JBOSS Fuse 6	● STIG for Apache ActiveMQ ● Standard System Security Profile for JBOSS ● STIG for JBOSS Fuse 6
Java Runtime Environment	● Java Runtime Environment (JRE) STIG
Oracle Linux 8	● Criminal Justice Information Services (CJIS) Security Policy ● Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171) ● [DRAFT] Australian Cyber Security Centre (ACSC) Essential Eight ● Health Insurance Portability and Accountability Act (HIPAA) ● Standard System Security Profile for Oracle Linux 8 ● ANSSI-BP-028 (enhanced) ● DRAFT – ANSSI-BP-028 (high) ● ANSSI-BP-028 (intermediary) ● ANSSI-BP-028 (minimal) ● [DRAFT] Protection Profile for General Purpose Operating Systems ● PCI-DSS v3.2.1 Control Baseline Draft for Oracle Linux 8
Red Hat Enterprise Linux 9	● ANSSI-BP-028 (enhanced) ● ANSSI-BP-028 (high) ● ANSSI-BP-028 (intermediary) ● ANSSI-BP-028 (minimal) ● [DRAFT] CIS Red Hat Enterprise Linux 9 Benchmark for Level 2 – Server ● [DRAFT] CIS Red Hat Enterprise Linux 9 Benchmark for Level 1 – Server ● [DRAFT] CIS Red Hat Enterprise Linux 9 Benchmark for Level 1 – Workstation ● [DRAFT] CIS Red Hat Enterprise Linux 9 Benchmark for Level 2 – Workstation ● Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171) ● [DRAFT] Australian Cyber Security Centre (ACSC) Essential Eight ● [RHEL9 DRAFT] Health Insurance Portability and Accountability Act (HIPAA) ● [RHEL9 DRAFT] Australian Cyber Security Centre (ACSC) ISM Official ● Standard System Security Profile for Red Hat Enterprise Linux 9 ● [DRAFT] DISA STIG with GUI for Red Hat Enterprise Linux 9 ● [RHEL9 DRAFT] Criminal Justice Information Services (CJIS) Security Policy ● [RHEL9 DRAFT] Protection Profile for General Purpose Operating Systems ● [RHEL9 DRAFT] PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 9 ● [RHEL9 DRAFT] Red Hat Corporate Profile for Certified Cloud Providers (RH CCP) ● [DRAFT] DISA STIG for Red Hat Enterprise Linux 9
SUSE Linux Enterprise 15	● CIS SUSE Linux Enterprise 15 Benchmark ● Standard System Security Profile for SUSE Linux Enterprise 15 ● PCI-DSS v3.2.1 Control Baseline for SUSE Linux enterprise 15 ● DISA STIG for SUSE Linux Enterprise 15

¹¹⁴ Choosing a Policy <http://www.open-scap.org/security-policies/choosing-policy/>

ターゲット	SCAP Security Guide
Ubuntu 20.04	● CIS Ubuntu 20.04 Level 1 Server Benchmark ● CIS Ubuntu 20.04 Level 1 Workstation Benchmark ● CIS Ubuntu 20.04 Level 2 Server Benchmark ● CIS Ubuntu 20.04 Level 2 Workstation Benchmark ● Standard System Security Profile for Ubuntu 20.04 ● Canonical Ubuntu 20.04 LTS Security Technical Implementation Guide (STIG) V1R1

SCAP Security Guide のコンテキストは最終的には xml フォーマットのデータストリーム (ds) ファイルとして提供される。GNU/Linux の商用サポート版では標準的にサポートされるが、コミュニティ版では必ずしも標準的にはサポートされていない。そのため公開されているデータやファイルからインストールするノウハウが必要である。

多くの場合、GNU/Linux ディストリビューターが自分たちのセキュリティポリシーのために用意しているが、それだけではなく DISA STIG (米国国防総省国防情報システム局の作成した Security Technical Implementation Guide) も用意されている。

4.1.2.2.2. 特定のソフトウェア向け SCAP Security Guide

SCAP Security Guide を定義することで特定のソフトウェアに対するセキュリティ監査を行うことも可能である。Firefox などは DoD 向けの SCAP Security Guide が用意されている。他にも OSS に関して DoD 向けの SCAP Security Guide には Apache や Bind もある。

DISA が DoD 向けに作成する STIG のように、このような特定の組織や団体、あるいは利用環境向けに `oscap` コマンドで利用可能な SCAP Security Guide を作成して配布することによって高いレベルで統一的なセキュリティポリシーの適用と、統一的なセキュリティ監査が可能になると考えられる。

OpenSCAP のツール群もまた OSS であり、ソースコードを入手し環境にあわせてインストールすることが可能である。また一部の GNU/Linux ディストリビューションでは `oscap` 以外にも `scap-workbench` や `openscap-daemon` などのツールも利用可能な場合がある。作成には、CVE、OpenSCAP だけではなく、対象となるシステムに対しての深いセキュリティ知識が必要なので求められるスキルが高い。これまで GNU/Linux のディストリビューターが作成し用意する、あるいは DISA が用意するなど供給元が極めて限られたが、これまで OSS が多様なニーズにあわせて発展していったように、SCAP Security Guide もオープンソースの 1 つとして発展し多様化していくことで、より OSS のセキュリティが強化される可能性は大きいと考えられる。

4.1.3. サプライチェーンにおける OSS 管理

サプライチェーンにおける OSS 管理に関連した取組みとして情報収集した結果を報告する。4.1.3.1 に各種のディストリビューションのメンテナンス方法を、4.1.3.2 に OpenChain プロジェクトを報告する。

4.1.3.1. Linux ディストリビューションのメンテナンス方法

ここでは GNU/Linux のディストリビューションにおける OSS のメンテナンスの流れと問題点を説明する。

GNU/Linux が多く利用されている分野の 1 つとして Web サーバーシステム（以下、Web サーバー）が挙げられる。GNU/Linux をプラットフォームとし、OSS である Web サーバーシステムの Apache や nginx が利用されている。netcraft が提供する 2021 年 10 月の調査データ¹¹⁵によれば、インターネット上でアクティブに公開されている Web サイトでのシェアは nginx（29.9%）、Apache（24.7%）であり、シェア率 1 位と 2 位となっている。このように現在のインターネットにおいて OSS は欠かせない要素となっている。

4.1.3.1.1. GNU/Linux のディストリビューション

Web 技術調査サイト W3techs の 2021 年 12 月の調査によれば¹¹⁶、インターネット上で Web サーバーを稼働させているサイトの OS は UNIX 系 OS が 78.8%を占めている。また、全体のうち GNU/Linux は 38.7%を占めている¹¹⁷。

GNU/Linux はディストリビューションと呼ばれるシステム形式を取っている。カーネル、ミドルウェア、アプリケーションが同じソフトウェアであっても、構成方法や管理方法が異なり、独自性を打ち出している。Red Hat、SUSE、Canonical、Oracle など企業が提供する各種ディストリビューション、米国非営利団体 Software in the Public Interest が運営する Debian のような法的エンティティ（公認された団体）が運営するディストリビューション、あるいはコミュニティがサポートするディストリビューションなど多用なディストリビューションが存在する。先の W3techs の調査によると GNU/Linux の Web サーバーのディストリビューション上位 3 つは Ubuntu（34.9%）、Debian（15.5%）、CentOS（9.9%）である。

Web サーバーを稼働させる場合は、ディストリビューションに標準で用意されているバイナリの実行形式ファイル、設定ファイル、関連ファイルなどからなるインストールパッケージをパッケージ管理システム経由でインストールするのが推奨される。一方で、OSS はソースコードが入手可能なのでユーザー独自にコンパイルし実行可能形式を作成しシステムにすることもできる。

この両者を脆弱性が発生した後のメンテナンスという面から比較してみると、前者はアップデートがディストリビューターから提供されるとすぐにパッケージ管理システムを利用してアップデート可能であるのに対して、後者は再コンパイルしなければならない。また後者は前者に比べて技術的なスキルを必要とし、作業量も多くなるので作業コストとしては大きくなる傾向がある。

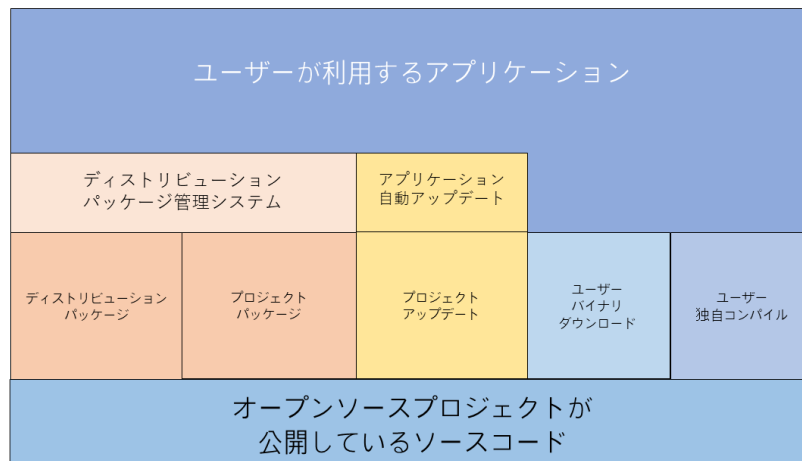
4.1.3.1.2. メンテナンスからみたディストリビューション

ユーザーが利用する OSS のアプリケーションのソースコードが修正された場合、どのような形でアップデートされたアプリケーションがユーザーまで届くか整理すると下図のようになる。

¹¹⁵ October 2021 Web Server Survey <https://news.netcraft.com/archives/2021/10/15/october-2021-web-server-survey.html>

¹¹⁶ October 2021 Web Server Survey <https://news.netcraft.com/archives/2021/10/15/october-2021-web-server-survey.html>

¹¹⁷ （注意）GNU/Linux は UNIX 系 OS としてカウントされている。



出所) 各種ディストリビューションの情報に基づき三菱総合研究所作成

図 4-3 ユーザーのインストール及びメンテナンスのモデル

表 4-4 配布される方法の種類とアップデートの流れ

配布される方法	アップデートの流れ
ディストリビューションパッケージ (標準パッケージ)	ディストリビューションのパッケージメンテナ (パッケージのメンテナンス担当者) がプロジェクトからソースコードを入手し、ディストリビューションに最適なディストリビューションパッケージを作成し、ディストリビューションから正式パッケージとして公開される。パッケージ管理システムによりユーザーにアップデートの指示が示されるか、もしくは自動的にアップデートされる。
プロジェクトパッケージ	プロジェクトのディストリビューターが標準パッケージと同じ手順でパッケージを作成し、オプショナルなパッケージとして公開する。標準パッケージと同様な方法でアップデートされる。
プロジェクトアップデート	アプリケーション自体が自動アップデート機能を持っており、アプリケーションを配布しているプロジェクトがアップデートを公開するとユーザーにアップデートの指示が示されるか、あるいは自動的にアプリケーションがアップデートされる。
ユーザーバイナリダウンロード	ユーザーが自分で配布サイトからダウンロードしてきてインストールもしくはアップデートを行う。
ユーザー独自コンパイル (ユーザービルド)	ソースコードもしくはソースコードパッチを自分でダウンロードし、独自にコンパイル環境 (ビルド環境) を作成しアプリケーションを作成しシステムにインストールする。

「OSS のアプリケーションをユーザーがアップデートする」ということであっても、OSS のソースコードにパッチが当てられてから、ユーザーにアプリケーションのアップデート版が届くまで、色々なルートが考えられる。OSS のメンテナンスやアップデートに関して議論する場合には、どのケースを想定して議論するのかを明確化する必要がある。例えば、OSS の Web ブラウザ Firefox を GNU/Linux のディストリビューションで利用する場合を考

えてみる。ディストリビューションの標準パッケージとして用意されているのでシステムのパッケージ管理システムが自動的にアップデートする。このアップデートのためのパッケージを作成しているのはディストリビューションのメンテナーである。Windows 上で利用している場合、Firefox が自動的にアップデートを検知しアップデートを行う。この場合、Firefox プロジェクトがアップデート用の Windows バイナリパッケージを作成し配布している。Firefox は OSS プロジェクトであるので、ユーザーが Firefox のソースコードを入手し独自にビルドすることも可能である。ただしこの場合、ユーザーは開発者レベルのスキルが必要である。「OSS が（脆弱性対応で）アップデート」といった時の議論をした時に、どの環境においてのアップデートなのかで、利用者のアップデートにかかるコストがまったく違ってくるので注意が必要である。

4.1.3.1.3. パッケージ管理システムによる管理の効率化

各ディストリビューションには主催する組織による運用も含めパッケージ管理をする包括的な仕組みが標準として用意されている。ソフトウェアの通常のアップデート以外にも緊急のセキュリティアップデート¹¹⁸も、その中に含まれる。ディストリビューション側で用意され、自動的にシステムからユーザーに通知される。これらの仕組みは Windows アップデートや macOS のソフトウェア・アップデートの機能と同様の仕組みである。そのためディストリビューションがサポートしている範囲の OSS を利用する限り、ユーザーはシステムのソフトウェア更新の仕組みにより脆弱性対応はユーザーが意識することなく対応できるようになっている。

表 4-5 パッケージ管理コマンドと採用しているディストリビューション

パッケージ管理コマンド	利用しているディストリビューション
rpm	Red Hat ほか
apt	Debian Ubuntu ほか
yum	CentOS Fedora ほか
dnf	yum の後継パッケージ管理コマンド
snap	Ubuntu (apt と共存)

これらのパッケージ管理コマンドは 1 つ、もしくは複数からなるコマンドで、システムに対してインストール、アンインストール、アップデートなど包括的なソフトウェア管理を行うことができる。端末からのコマンド入力を行う以外にも GUI インタフェースを持つアプリケーションとしても用意されている。ディストリビューションにアップデートが用意されると、ユーザーに対してアラート/メッセージが表示し推奨されるアップデートを提示する。ユーザーは、そのメッセージ/アラートに従ってアップデートを行う。設定によっては、自動的にアップデートを行うことも可能である¹¹⁹。

利用しているソフトウェアの依存関係はパッケージ管理コマンドで管理されており、どのような依存関係があるのかユーザーは意識することなくセキュリティアップデートも含

¹¹⁸ 脆弱性などの対応のアップデートをクオリティアップデート（QU）、機能の向上のためのアップデートをフューチャーアップデート（FU）と区別する場合もある。

¹¹⁹ （参考）CentOS（RedHat 系）の yum-cron

めアップデートが行われる。このようにパッケージに含まれる既知の脆弱性を修正しアップデートを配布するのがディストリビューションの役目の 1 つであり、ディストリビューションの標準環境の範囲ではスムーズにアップデートが行われる。一方で、標準環境ではなくユーザー独自のシステム環境として拡張、あるいは構築している場合、ディストリビューターが行っているアップデート作業と同等な作業（情報収集も含む）をユーザーが行うことになるため、相応のスキルを要求される。

4.1.3.2. OpenChain プロジェクト(ISO/IEC 5230:2020)

OpenChain ISO/IEC 5230:2020（以下 OpenChain）は、オープンソース・ライセンスのコンプライアンスに関する国際規格である。OpenChain プロジェクトは OpenChain の普及・推進を目指した取り組みで The Linux Foundation が主催している。OpenChain プロジェクトの会員資格には、The Linux Foundation の会員企業が参加できる有償のプラチナ会員と、誰でも参加できるコミュニティ会員がある¹²⁰。

4.1.3.2.1. OpenChain のセキュリティへの拡張

現在は、サプライチェーンにおけるセキュリティの確保が注目され、オープンソース・ライセンスのコンプライアンスから、セキュリティへと重心が移っている。OpenChain ワーキンググループは、CVE、GitHub/GitLab、パッケージマネージャーなどからもたらされる脆弱性情報を取り入れて OSS を精査するための枠組みへとシフトしている。ISO/IEC 5230:2020 の内容は変えないが、その内容をセキュリティとして読み替えるためのガイダンス OpenChain Security Assurance Reference Guide 1.0 が用意されている^{121,122}。以下に、例としてそれぞれの 3.1.1 と 3.1.5 の記載の違いを明記する。

表 4-6 OpenChain ISO/IEC 5230:2020 3.1.1 と 3.1.5 の相違点

OpenChain ISO/IEC 5230:2020 オリジナルの記載 ¹²³	OpenChain Security Assurance Reference Guide 1.0 の拡張 ¹²⁴
3.1.1 Policy A written open source policy shall exist that governs open source license compliance of the supplied software. The policy shall be internally communicated.	3.1.1 Policy A written policy shall exist that governs Open Source Security Assurance of the Supplied Software. The policy shall be internally communicated.
3.1.5 License obligations A process shall exist for reviewing the identified licenses to determine the obligations, restrictions and rights granted by each license	3.1.5 Standard Practice Implementation ● Organization knowledge of Known Vulnerabilities exists ● Method for detecting existence of Known Vulnerabilities in Supplied Software

¹²⁰ OpenChain <https://www.openchainproject.org/>

¹²¹ OpenChain Security Assurance Reference Guide 1.0 <https://www.openchainproject.org/security-guide>

¹²² OpenChain ISO/IEC 5230 -Security Assurance Reference Guide（日本語版）：
https://qiita.com/kida_oss/items/bf6a9d005dd1b50b5875

¹²³ INTERNATIONAL STANDARD ISO/IEC 5230, ISO/IEC 5230:2020(E) First edition, 2020-12 Information technology — OpenChain Specification

¹²⁴ OpenChain Security Assurance Reference Guide 1.0 <https://www.openchainproject.org/security-guide>

OpenChain ISO/IEC 5230:2020 オリジナルの記載 ¹²³	OpenChain Security Assurance Reference Guide 1.0 の拡張 ¹²⁴
	● 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 ● Organization knowledge of Known Vulnerabilities exists ● 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

4.1.3.2.2. OpenChain プロジェクトと連動するプロジェクト

OpenChain プロジェクトと連動するプロジェクトとして、Open Source Security Foundation (OpenSSF) プロジェクトと Software Package Data Exchange Project (SPDX) プロジェクトの 2 つがある。OpenSSF はオープンソースをより安全にするための共同体であり、The Linux Foundation の姉妹組織として位置づけられている。SPDX プロジェクトは Software Package Data Exchange (ISO/IEC 5962:2021) を開発し、現在は保守しているプロジェクトである。SPDX は OpenChain を実施する際の情報交換フォーマットとして開発・発展してきたもので、現在は ISO/IEC 5962:2021 として国際標準となっている。

4.1.4. 脆弱性情報等の共有及び共有体制

脆弱性情報の共有は、国際的には、CVE 及び NVD が存在し、国内では JVN 等がある。また、脆弱性情報の共有体制には、これら以外に CVE や CAN が存在する。OSS を利用する際や脆弱性情報の収集・共有には参考となるため、以下にこれらを概説する。

4.1.4.1. CVE 及び NVD

4.1.4.1.1. CVE プログラム

当初は CVE という認識であったが、現在では「CVE プログラムは公開されているサイバーセキュリティの脆弱性を特定、定義、カタログ化することを目的としている」とした包括的な枠組みで捉えている¹²⁵。発見された脆弱性に一意な CVE 番号を割り付け管理すること

¹²⁵ CVE サイト <https://www.cve.org>

は、脆弱性情報の流通化、カタログ化、また脆弱性ハンドリングに対して対象を明確化できる利点がある。これまで CVE プログラムは米国連邦政府（DHS CISA）の資金提供をうけて米国非営利団体 MITER が採番、及び情報管理をしてきたが、現在では CVE プログラムへ参加している組織が世界各地に増え、脆弱性情報の管理が分散化されている。CVE プログラムに参加する組織の分類を大別すると CAN（CVE Numbering Authority）と Root CNA の 2 つになる。CNA は脆弱性に対して脆弱性番号を付与しカタログ化する組織である。Root CNA は管轄内の CNA やステークホルダーとの調整など CNA のまとめ役的な組織である。2022 年 1 月 1 日現在、Root CNA と CNA の総数は 209 組織であり、うち Root CNA は 4 組織である¹²⁶。

表 4-7 Root CNA 一覧 (2022/01/05 時点)

組織名	対象範囲	役割	組織タイプ	国
MITRE Corporation	All vulnerabilities, and Open Source software product vulnerabilities, not already covered by a CNA listed on this website	Top-Level Root, CNA, CNA-LR, Secretariat	N/A	USA
Cybersecurity and Infrastructure Security Agency (CISA) Industrial Control Systems (ICS)	Industrial control systems and medical devices	Top-Level Root, CNA-LR, CNA	National and Industry CERTs	USA
JPCERT/CC	Root Scope: Japan organizations CNA Scope: Vulnerability assignment related to its vulnerability coordination role	Root, CNA	National and Industry CERTs	Japan
Spanish National Cybersecurity Institute, S.A. (INCIBE)	Spain organizations CNA Scope: Vulnerability assignment related to its vulnerability coordination role for Industrial Control Systems (ICS), Information Technologies (IT), and Internet of Things (IoT) systems issues at the national level, and vulnerabilities reported to INCIBE by Spain organizations and researchers that are not in another CNA's scope	Root, CNA	National and Industry CERTs	Spain

4.1.4.1.2. NVD

NVD は米国政府の脆弱性データベースである¹²⁷。CISA が資金提供し NIST Computer Security Division が運営している。CVE 番号が割り振られ公開される脆弱性に対して分析を行い、脆弱性の説明作成、ベンダーが提供している脆弱性情報などの参照先の整理、その他にも公開されている補足情報やサイト情報などの集約を行う。分析の範囲には関連付

¹²⁶ List of Partners <https://www.cve.org/PartnerInformation/ListofPartners>

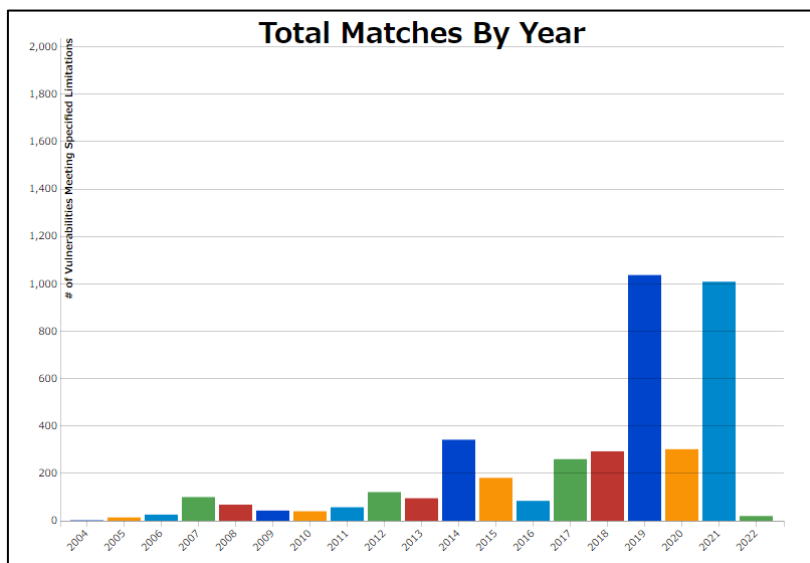
¹²⁷ General Information <https://nvd.nist.gov/general>

けの影響メトリック（CVSS）、脆弱性タイプ（CWE）、適用性ステートメント（CPE）、及びその他の関連するメタデータが含まれる。それらの脆弱性情報はNVD サイト上のコンテンツとして一般公開されているだけではなく、SCAP を用いて脆弱性情報を発信する。それにより脆弱性情報管理、影響度、セキュリティ監査などの自動化が可能になっている。NVD は 2000 年に作成された ICAT（Internet-Categorization of Attacks Toolkit）からスタートし、これまで何度も変遷を重ねて現在の形になっている。2021 年末でデータベースのエントリー数は約 15 万件となっている。

4.1.4.1.3. OSS における CVE/NVD の傾向

CVE/NVD に登録される脆弱性はソフトウェア/システムの影響度を加味している。脆弱性により広く影響、あるいは深刻な影響があるようなケースであれば CVE 番号の取得が可能であるが、利用者が限られているようなケースの場合、CVE 番号の取得ができないケースがある。OSS と一口にいっても Github 上で趣味的に公開しているソースコードから Linux カーネルのように情報基盤として利用されているものまで幅は広い。OSS でも利用者が限られ影響範囲が狭いケースがあるため、脆弱性情報が寄せられても脆弱性管理の枠組みで処理されない脆弱性となる可能性がある。そのため CVE 番号があるものに関しては対処が必要な脆弱性であるといえるのは当然であるが、CVE 番号がついていないから対処が必要ではない脆弱性とはならないので注意が必要である。

また、CVE/NVD への登録数が多く、NVD の分析が追いつかないケースもある。例えば、Wordpress のプラグインに関してはプラグインの提供数増加と、そこでの脆弱性発見のペースが上がっており、NVD の分析が追いつかず CVE/NVD 公開時に CVSS の値が用意できないケースも散見される。そのため脆弱性の影響度が CVSS を参考に判断できないという場合もあるので注意が必要である。CVSS の値の大小は CVSS が用いているメトリクスであって、参考値として利用することは意味があるが、実際に利用している個別のシステムの利用環境で特性を前提にした脆弱性の影響を考慮すべきである。



出所) NVD <https://nvd.nist.gov/vuln/search>

図 4-4 Wordpress に関連する CVE 発行数の統計

4.1.4.2. 日本国内の共有体制及び JVN

4.1.4.2.1. 国内における脆弱性情報共有体制

2004 年 7 月 8 日から経済産業省の告示に基づき策定された情報セキュリティ早期警戒パートナーシップガイドラインに則り、脆弱性関連情報の届出受付機関は独立行政法人情報処理推進機構（IPA）、脆弱性関連情報に関して製品開発者への連絡及び公表に係る調整機関として一般社団法人 JPCERT コーディネーションセンター（JPCERT/CC）という位置づけになっている。脆弱性に関連する情報を関係者間で調整後、JPCERT/CC、IPA からの告知の他に JVN サイトでの公開を行う。JVN は JPCERT/CC と IPA が共同で運営している。

日本で使用されているソフトウェアなどの脆弱性関連情報とその対策情報を提供し、情報セキュリティ対策に資することを目的とする脆弱性対策情報ポータルサイトです。脆弱性関連情報の受付と安全な流通を目的とした「情報セキュリティ早期警戒パートナーシップ」に基づいて、2004 年 7 月より JPCERT コーディネーションセンターと独立行政法人情報処理推進機構(IPA)が共同で運営しています。

(JVN サイトより引用¹²⁸⁾)

ここで扱う脆弱性情報とは大きく分類して、製品に対する脆弱性と、運用されている Web サイトに存在している脆弱性の 2 つあり、本章の説明では製品に対する脆弱性の流通についてのみ取り上げる。また製品の脆弱性では設定に起因する脆弱性ではなく（例えば、デフォルトパスワードが利用可能になっている）ではなく、ソフトウェアの瑕疵を起因とする脆弱性について述べる。

4.1.4.2.2. 製品の脆弱性のコーディネーション

製品の脆弱性のコーディネーションは JPCERT/CC が中心となり、コーディネーションを行う（図 4-5）。なお、現在では JPCERT/CC が CVN 採番可能な Root-CNA 組織として登録され、また JPCERT/CC 傘下に sub-CNA という形で CVN 採番が可能な組織が作られているので図とは細かい点で若干の相違はあるが、基本的な流通体制は図 4-5 のようになっている。

¹²⁸ JVN とは？ <https://jvn.jp/nav/jvn.html>

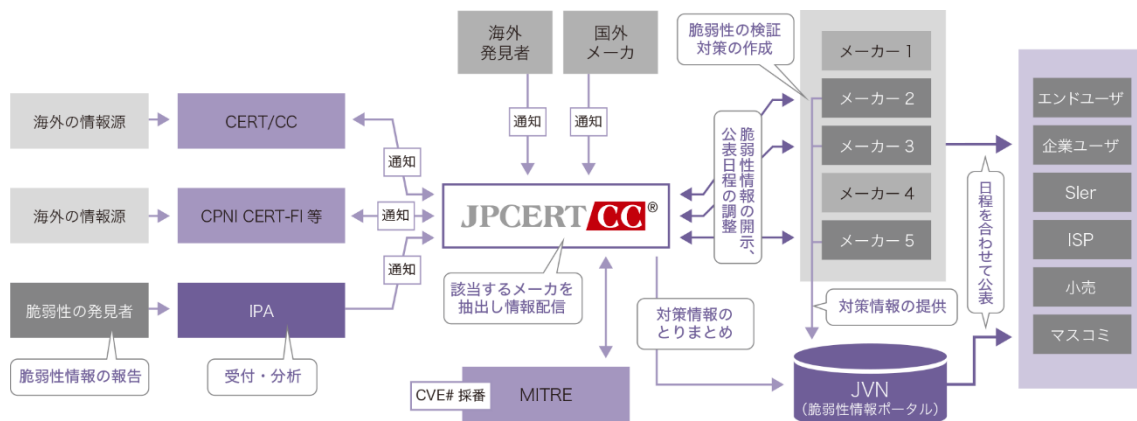


図 4-5 JPCERT/CC からみた脆弱性ハンドリング（JPCERT/CC サイトより¹²⁹⁾

4.1.4.2.3. 脆弱性情報の公開のタイミング

通常の場合、脆弱性情報の公開タイミングは次のようになる。

- 脆弱性を解消した修正版のソフトウェアを提供する状態になった時
 - （ケース 1）ベンダーから安全になったソフトウェアを提供できるタイミングで脆弱性情報を公開しアップデートなどを推奨する。
- 脆弱性を解消することができなかった時
 - （ケース 2）ベンダーや脆弱性対応組織から脆弱性の影響を緩和するための方法を公開し推奨する。
 - （ケース 3）ベンダーや脆弱性対応組織から脆弱性の問題を解決できないとして、利用を中止することを推奨する。

ケース 1 及びケース 2 の場合は、脆弱性を含んでいるソフトウェアを利用するユーザーに対しての不利益がないように対応と同時に脆弱性情報を公開することが可能となった、脆弱性ハンドリング・コーディネーションが一定の成果を示したケースである。

ケース 3 の場合、対象となるソフトウェアが例えば End-of-Life となっており、ベンダーが正式にアップデートや修正を行うことを行わず利用を中止することを推奨する場合などは脆弱性情報のコーディネーションを行うことができたといえる。しかし、ベンダーが脆弱性を認めず（例えば、仕様であると主張）コーディネーションが不調に終わるケースが考えられる。このまま脆弱性を放置することで被害が発生すると脆弱性対応組織が判断した場合は、ベンダーが脆弱性を認めなくとも利用を中止することを推奨する場合があります。またベンダーに対して脆弱性情報を提供しようとしてコンタクトをしようとしてもコンタクト先が不明である、あるいはコンタクトしても反応がない（無視される）場合においても、利用を中止することを推奨する場合があります。CISA など米国の脆弱性対応機関では、まったくコンタクトができない場合（コミュニケーションを拒む場合も含む）、ベンダーに対して脆弱性報告を試みた時点から 45 日を目処に公開するとされている（45 日ルール）。

¹²⁹⁾ JPCERT/CC 事業概要・脆弱性情報ハンドリング https://www.jpccert.or.jp/about/06_3.html

日本国内においてコーディネーションが不調に終わった脆弱性に関しては、脆弱性届出窓口である IPA において調整不能案件という位置づけで扱い、定期的に行われる公表判定委員会の判定を経て公表されるという、慎重なプロセスを経て公開される。

4.1.4.2.4. OSS における脆弱性対応の課題

プロプライエタリ・ソフトウェアでは、ライセンスで第三者による改造などを許していないため、技術的に修正が可能・不可能に関わらず、ベンダーと脆弱性対応組織との関係で対応することになる。

OSS の場合、誰でもソースコードを入手し改良することが可能であるソフトウェアという性質から、プロプライエタリ・ソフトウェアよりも対応が複雑になりうる。開発者にコンタクト不可能な場合や、あるいは何らかの理由で開発者とコーディネーション不調で終了する（利用しないことを推奨する）場合であっても、脆弱性対応が行える場合がありうる。以下に想定されるケースを挙げる。

表 4-8 脆弱性対応の想定ケース

想定ケース No	概要
想定ケース 1 ディストリビューションのメンテナが対応する場合	GNU/Linux などのディストリビューションに含まれるソフトウェアの場合、そのソフトウェアのメンテナ（メンテナンス担当者/世話役）が存在する。ソフトウェアの開発者がなんらかの理由で脆弱性に対応できない場合、メンテナがソースコードを修正し、ディストリビューションのソフトウェアのアップデートを行う場合がある。オリジナルの開発者にコンタクトが取れなかったとってコーディネーション不調として処理するのではなく、それらのアップデート情報を脆弱性対応組織はキャッチアップする必要がある。
想定ケース 2 利用者が直接対応する場合	ここでの利用者ではエンドユーザーではなく、OSS をソフトウェア製品の一部もしくはそのものを利用したシステムを提供するベンダーであるとする。既に OSS 開発コミュニティでメインストリームのバージョンではなく、かつ、既にメインストリームに移行できないような旧バージョンのソフトウェアをシステムに組み込んでいるような製品を想定する。そのような製品に新たな脆弱性を利用者が発見してしまった場合、旧バージョンのソフトウェアの脆弱性情報を OSS 開発コミュニティに提供しても、既に OSS 開発コミュニティでは End-of-Life となっているので修正はかからない場合がある。製品の脆弱性対応を行うならば、利用者が OSS の旧バージョンのコードに対して自ら修正をかけ脆弱性対応を行い、ベンダーとして製品のアップデートリリースを行う必要がある。修正した脆弱性情報に関しては脆弱性対応組織に届出を行うことによりユーザー以外で脆弱性を認識することになる。利用している OSS のライセンスにもよるが、修正したコードは同じく OSS のライセンスとして公開する必要性が生じることも考慮に入れる必要がある。
想定ケース 3 オープンな修正	脆弱性対応が既に終了し、対応したバージョンを提供すると共に脆弱性情報を公開する標準的なプロセスでコーディネーションが一旦終了したとする。しかし修正したバグ（ここでは脆弱性を含むソフトウェアの瑕疵であるかどうかは問わない）だけではなく、複数のバグが存在していてその修正漏れのまま公開してしまうケースがある。あるいは修正コードを入れたために、その影響で別のところで違うバグが出てしまうケースもある。ソースコードは公開されているので、世界中のプログラマが次々とバグを見つけては修正するバグスマッシング（もぐらたたき）状態になるケースは多くはないが、そのような状況になる時がある。すべてのバグが脆弱性ということではないが、バグの中には脆弱性も含まれる場合もある。2014 年に発生した bash の一連の脆弱性（CVE-2014-6271 ほか）がこのケースであった。その場合、発見されるバグと次々に修正されていくコードに対して、バグは脆弱性となりうるのか否か、また修正コードは脆弱性の修正となっているのか否かをキャッチアップしていく必要がある。公開の場で行われるが、修正されるバグが脆弱性となりうるのか否かは、容易に判断することができない、あるいは容易に検証できないケースもある。一方で、脆弱性の場合、対処情報よりも先に、その脆弱性の存在が公開されてしまうことになる

ケース 3 などのように OSS は脆弱性が発見された場合、公開された場で関係者を巻き込むことで短い期間で多くのバグを解決することができる可能性がある良い面と、脆弱性情報がコーディネーションされず（あるいは、誰がコーディネーションするのが曖昧）に公開される可能性があるという悪い面が同時に現れるケースがありうる。OSS に対する脆弱性対応のハンドリングやコーディネーションは OSS の特性からプロプライエタリ・ソフトウェアを提供するベンダーへの対応よりも多くのステークホルダーや第三者の開発者を加

えた複雑になるケースを想定し、柔軟に対応しなければならないことが分かる。

4.1.4.3. CVE と CNA の関係及び Sub-CNA の役割

CVE¹³⁰プログラムは公開されている脆弱性を特定、定義、カタログ化するための枠組みで、脆弱性に対しての識別子として CVE 番号を割り当て管理する。CVE 番号は脆弱性に対するハンドリング対象の明確化、カタログ化、脆弱性情報が流通する際のインデックスとして利用できるように 1 つの脆弱性に 1 つ割り当てている。

4.1.4.3.1. CNA と Sub-CNA

脆弱性情報に対して CVE 番号を付与しカタログ化する組織が CNA である。当初、CNA は米国非営利組織 Mitre Corporation のみであったが、脆弱性の増加などの要因により 2010 年前後から分散化が進んだ。その後、Adobe、Cisco Systems、Microsoft といった大手 IT ベンダー、CSIRT 組織からは CERT/CC と JPCERT/CC が CNA として加わっていた。

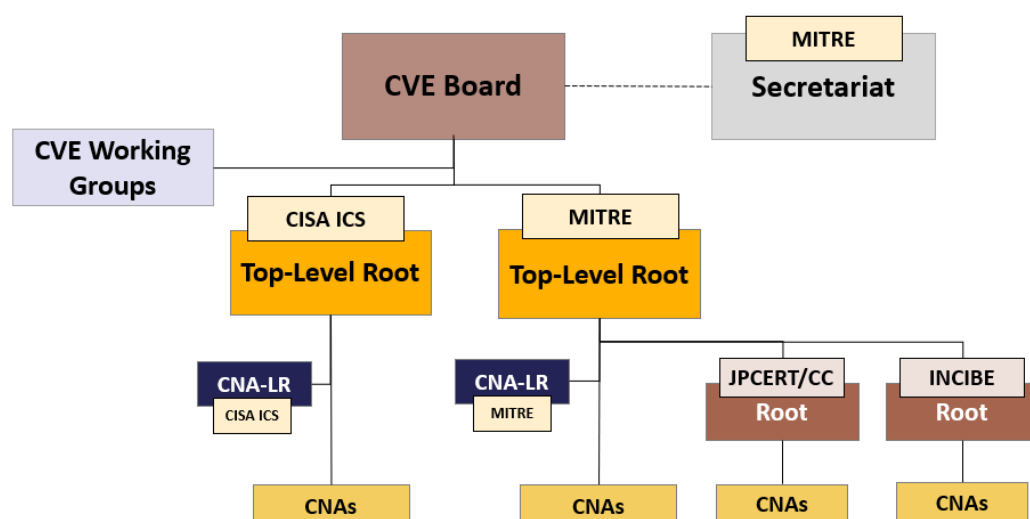


図 4-6 CNA の構造 (MITRE Corporation¹³¹より)

その後、新しい CNA 組織は増加し、現在は、Top-Level Root、Root、CNA、CNA-LR という区分けされた階層構造になっている。Top-Level Root は CISA ICS と MITRE の 2 組織となっている。CNA-LR は CNA からの CVE 番号割り当ての調整を行い、現在は CISA ICS と MITRE となっている。Root は、その傘下に複数の Sub-CNA があり、その傘下の CNA に対する調整を行う。最終的には CVE Board が CVE の全体調整を行う仕組みになっている。

2022 年 1 月末において CVE プログラムに参加している組織数は約 209 組織、参加国は約 34 カ国となっている。

¹³⁰ 本報告書「2.1.3.4.1 CVE 及び NVD」に関連情報

¹³¹ Structure <https://www.cve.org/ProgramOrganization/Structure>

表 4-9 CVE プログラム参加組織数

組織種類	登録組織数
CNA	209
CNA-LR	2
Root	3
Top-Level Root	2

(注:CISA ICS や MITRE などは複数カウントされているため表の合計が参加組織数とはならない)

CVE プログラムの参加組織の種類を以下に示す。

表 4-10 CVE プログラム参加組織の種類

種類	組織数
バグバウンダリプログラム	7
ホストサービス	3
CERT 組織	11
ベンダー・プロジェクト	184
脆弱性研究組織・研究者	40

CVE プログラムの参加組織の国別（上位 8 ヶ国）を以下に示す。

表 4-11 CVE プログラム参加組織の国別（上位 8 ヶ国）

国	参加組織数
米国	111
中国・ドイツ	9
日本	8
カナダ・スイス・台湾・英国	6

4.1.4.3.2. Root CNA としての JPCERT/CC と国内 Sub-CNA

JPCERT/CC が Root CNA として傘下に収める Sub-CNA は、東芝、パナソニック、NEC、三菱電機、LINE となっている¹³²。これらの組織の CNA の役割（対象範囲）として、自社製品に対する脆弱性に対して CVE 番号を割り当てることができる。JPCERT/CC は、その上位 CNA 組織として調整にあたる。日本は電気・電子情報関連産業が盛んな国の一つであり、今後も JPCERT/CC 傘下には新規の CNA がさらに加わっていくことが予想される。

ベンダーが自前の CNA を持つ利点は、CVE 対応としてベンダーが自社製品に対する脆弱性の確認・検証を行うことができる点である。CVE 番号は一意に脆弱性を示す必要があり、脆弱性の検証・確認、また脆弱性の切り分けなどを外部の CNA と調整を行うことになることと作業のオーバーヘッドが大きくなることが考えられるが、自ら製品に対する CNA となることでオーバーヘッドを減少することが期待できる。

¹³² 2022 年 1 月末現在

LINE は自社の製品及びサービスに対して脆弱性の発見を公募し報告者に報奨金を支払う「LINE Security Bug Bounty Program」を2016年6月2日より実施している¹³³。自社サービスや製品に対する脆弱性の情報を積極的に収集する役割を自社内に持たせ、さらに CNA 機能を持たせることで脆弱性情報管理のオーバーヘッドを最小限にし、かつ一貫した対応ができる利点があると考えられる。

4.1.4.3.3. OSS と CNA の関係について

リーガルエンティティを持つ OSS プロジェクトとして CNA として登録できるクライテリアをクリアできる場合は、ベンダー同様に CNA となることも可能である。Web サーバーだけではなくインターネット上で使われる数々のソフトウェアを開発している Apache プロジェクトは、米国非営利団体¹³⁴Apache Software Foundation によって運営されているが、そこが主体となって CNA として参加している。また Ubuntu GNU/Linux のように Canonical が開発の主催・主幹をしている OSS コミュニティでは、ソフトウェアは OSS ライセンスで提供されていても、主催会社が CNA として参加しているものもある。GNU/Linux の有力なディストリビューション・ベンダーである Red Hat、SUSE など同様に CNA となっている。なお、OSS プロジェクトの Root CNA は現状では MITRE となっている。以下にいくつか例を挙げる。

表 4-12 関連する組織例

参加組織名	範囲	種別	組織種別	国
Apache Software Foundation	Apache ソフトウェアファンデーション関連のみ	CNA	非営利団体	米国
Canonical	Canonical 製品及び Ubuntu Linux	CNA	商用ベンダー	英国
Debian GNU/Linux	Debian のみ	CNA	非営利団体	米国
Document Foundation	ドキュメントファンデーションの製品及びサービスのみ (e.g. LibreOffice などセキュリティに関連するもののみ)	CNA	非営利団体	ドイツ
OpenSSL Software Foundation	OpenSSL プロジェクトのみ	CNA	非営利団体	米国

Apache Software Foundation/Apache プロジェクトはインターネット上で広く使われている Web サーバー Apache HTTP Server だけではなく、数々の OSS を提供している。2021 年 12 月 10 日に公開された Apache Log4j の任意のコード実行の脆弱性(CVE-2021-44228)の CNA は Apache Software Foundation であった。

ライセンスがオープンソース・ライセンスもしくはフリーソフトウェア・ライセンスと認定できるソフトウェアは FLOSS と定義できるが、一口に OSS といっても OSS を開発する

¹³³ LINE が CVE Numbering Authority (CNA) の一員に <https://linecorp.com/ja/security/article/355>

¹³⁴ 501(c)(3)団体として米国で登録されている

主体は多様である。その中でも開発主体・主催する組織が、会社や非営利団体としてリーガルエンティティを持ち、かつ脆弱性ハンドリング対応可能であるような OSS 開発プロジェクトは商用ベンダー同様に CNA となることが可能であり、また実際に複数の OSS プロジェクトが CNA として登録されている。

4.1.5. OSS コミュニティにおける連携・脆弱性流通の現状

OSS コミュニティにおける連携の取組みとして OSS 脆弱性管理支援プロジェクトである OSV を 4.1.5.1 に、OSS の脆弱性流通の現状の取組みとしてバグバウンティプログラムと脆弱性情報流通を 4.1.5.2 に報告する。

4.1.5.1. OSS 脆弱性管理支援プロジェクト OSV

4.1.5.1.1. OSV とは

Open Source Vulnerabilities (OSV) は、「オープンソースのメンテナーとオープンソースの利用者の両方を支援することを目的としたオープンソース・プロジェクトの脆弱性データベース及びトリアージ・インフラストラクチャ」を目指すオープンソース・プロジェクトである¹³⁵。スポンサーは Google である。

4.1.5.1.2. 「知る、防ぐ、修正する」のフレームワーク

OSS は、ソースコードがオープンなので依存関係を検証することが可能であり、それによりサプライチェーン攻撃に対するリスクを軽減できるように思えるが、それは十分な検証コストをかけることができるという前提があって始めて成り立つ。OSS は依存関係が多いという特徴があるため、OSS パッケージに含まれる、あるいは依存している OSS をすべて検証し、また、その後も検証し続けるコストを維持できるという仮定は非現実的であるといえる¹³⁶。

そこで Google は脆弱性情報のメタデータや IDなどを標準化し、脆弱性情報を共有すると共に脆弱性に対するワークフローの標準化・自動化することでソフトウェアの更新に必要な労力を軽減させ、脆弱性の影響を小限に抑えるというアプローチを 2021 年 2 月に Google Open Source Blog で公表し、OSV という形で公開した。その際に「Know, Prevent, Fix (知る、防ぐ、修正する)」というコンセプトを提唱している。

¹³⁵ Database for open source vulnerabilities <https://osv.dev/>

¹³⁶ Google Open Source Blog Know, Prevent, Fix: A framework for shifting the discussion around vulnerabilities in open source <https://opensource.googleblog.com/2021/02/know-prevent-fix-framework-for-shifting-discussion-around-vulnerabilities-in-open-source.html>

表 4-13 OSV のコンセプト概要

項目	概要
Know (知る) : ソフトウェアの脆弱性について知る	● 正確な脆弱性データを取得する。 ● 脆弱性データベースの標準スキーマを構築する。 ● 依存関係の正確な追跡が可能になるようにする。
Prevent (防ぐ) : 新しい脆弱性の追加を防ぐ	● 新しい依存関係のリスクを理解する。
Fix (修正する) : 脆弱性を修正又は削除する	● 脆弱性を削除するためのオプションを理解する。 ● 修正をスピードアップするための通知を行う。 ● 広く使用されているバージョンを修正する。

依存関係とその追跡は特に重要視している。Google は次のように説明している¹³⁷。

「プログラムが直接的又は間接的に何千ものパッケージやライブラリに依存することはよくあることです。例えば、Kubernetes は現在約 1,000 個のパッケージに依存しています。オープンソースは、クローズドソースよりも依存関係を利用している可能性が高く、幅広いサプライヤーからのものです。信頼する必要のある個別のエンティティの数は非常に多くなる可能性があります。これにより、製品でオープンソースがどのように使用されているか、及びどのような脆弱性が関連している可能性があるかを理解することが非常に困難になります。また、構築されたものがソースコードと一致するという保証もありません。(出典:Google Open Source Blog / Know, Prevent, Fix: A framework for shifting the discussion around vulnerabilities in open source)」

このように OSS のサプライチェーン管理は重要であるが、これまで脆弱性という面からサプライチェーンを管理するフレームワークがないため新たな枠組みを作る必要性を強調している。

また既存の CVE のようなフレームワークについては次のように説明している。

「オープンソース・ソフトウェアの利用者にとって、Common Vulnerabilities and Exposures (CVE) エントリーなどの脆弱性を、使用しているパッケージバージョンにマッピングすることはしばしば困難です。これは、既存の脆弱性標準 (Common Platform Enumeration (CPE) など) のバージョン管理スキームが、実際のオープンソースバージョン管理スキーム (通常はバージョン/タグ及びコミットハッシュ) とうまくマッピングされないという事実に起因します。その結果、下流の消費者に影響を与える脆弱性が見落とされます。(出典:Google Open Source Blog / Launching OSV - Better vulnerability triage for open source)」

¹³⁷ Google Open Source Blog Launching OSV - Better vulnerability triage for open source
<https://opensource.googleblog.com/2021/02/launching-osv-better-vulnerability.html>

4.1.5.1.3. OSV の脆弱性情報提供サイト osv.dev

Google は、脆弱性情報のメタデータのフォーマット（スキーマ）を定義し¹³⁸、そのフォーマットで交換される情報を利用するツール群を作り Github で公開し、メタデータの収集と公開を行うサイト osv.dev¹³⁹を運営している。Google は以前より OSS-Fuzz プロジェクトでファジング技術により OSS プログラムのバグ（含む脆弱性）を発見・集積・公開しているが、OSV のスタート段階では OSS-Fuzz からの脆弱性情報（メタデータ）を扱っていた。Google によれば OSS-Fuzz と統合された 380 以上の重要な OSS プロジェクトからの何千もの脆弱性を取り扱っていたとされる。最初、OSS-Fuzz（OSV）からの脆弱性情報を OSV 番号として扱った。その後、対応範囲を Go、Rust、Python、DWF（DWF は OSS 向けの CVE オルタナティブとして進められているプロジェクト¹⁴⁰）と発表した。現在は、OSS 向けの CVE オルタナティブ DWF ではなくグローバルな CVE/NVD ベースの GSD（Global Security Database）となっている。さらに GitHub Advisory Database（GHSA）が加わっている。このように GO や PYSEC のような開発コミュニティベースで採番されている脆弱性情報も集める。これまでの CVE のような包括的かつ統一的に脆弱性情報流通を管理している中央集権的な手法より OSS の開発がそうであるように、脆弱性情報に関してもコミュニティベースに移行すべきであるという考え方が背景にある。現在では、CVE も中央集権的な形から、Root-CNA や Sub-CNA という形で多様なバックグラウンドを持つ組織を増やし、かつ作業の分散化を図っている過程にあるので、当初の状況よりは変化している。まだ OSV も歴史が浅く、今は参加コミュニティを増やしている過程であり、まだ使えるツールも限定的であるので、今後どのような発展をしていくか、有用性はどうか、CVE との関係性はどうかといったことに関しては今後の課題である。

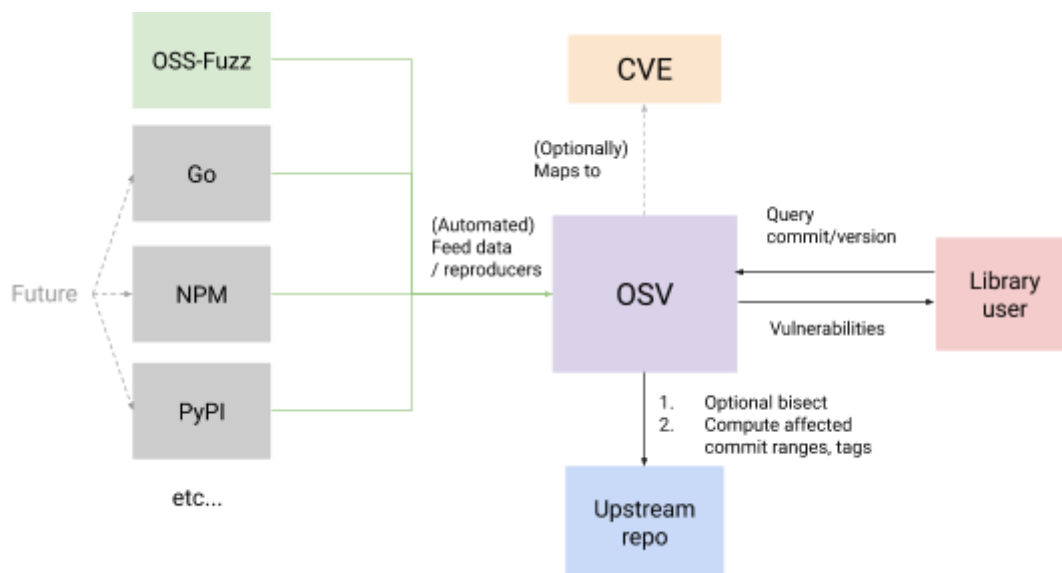


図 4-7 OSV のワークフロー（出典：Google Open Source Blog / Launching OSV - Better vulnerability triage for open source¹⁴¹）

¹³⁸ Open Source Vulnerability format Version 1.2.0 (January 19, 2021) <https://ossf.github.io/osv-schema/>

¹³⁹ <https://osv.dev/>

¹⁴⁰ <https://opensourcesecurity.io/2021/03/30/its-time-to-fix-cve/>

¹⁴¹ <https://opensource.googleblog.com/2021/02/launching-osv-better-vulnerability.html>

4.1.5.2. バグバウンティプログラムと脆弱性情報流通

4.1.5.2.1. バグバウンティプログラム例

バグバウンティプログラム（Bug Bounty Program）とは第三者によるソフトウェアの瑕疵（バグ）の発見報告に対し報奨（金）を与える制度である。バグバウンティそのものは1980年代から行われているが、近年では特に脆弱性を届け出た報告者に対して報奨金を与える制度のことを指すようになってきている。Microsoft、Facebook、Googleといった業界最大手企業らが独自のバグバウンティプログラムを用意している。日本国内でもサイボウズ、LINEといった会社が行っている。

表 4-14 バグバウンティプログラム例（Microsoft Corporation）

企業	Microsoft Corporation
名称	Microsoft Bug Bounty Program
報 償 内 容など	● クラウド認証系脆弱性に対して最大 10 万ドル ● Hyper-V 関連の脆弱性に対して最大 25 万ドル ● Windows 最新バージョンの認証回避などに対して最大 10 万ドル（さらに重要性を加味して 10 万ドル加算が可能） URL: https://www.microsoft.com/en-us/msrc/bounty

表 4-15 バグバウンティプログラム例（Meta Platforms, Inc.）

企業	Meta Platforms, Inc. (Facebook, Inc.)
名称	Hacker Plus
報 償 内 容など	● 報奨金となる製品もしくはサービスは次のとおり。 ➢ Facebook、Messenger、Instagram、WhatsApp、Quest、Workplace、Portal、Novi、Internet.org/Free Basics、Express Wi-Fi、Ray-Ban Stories、Meta によるオープンソース・プロジェクト ● 目安として 20 万人を超える Facebook ユーザーに影響が及ぶ可能性のある場合にのみ報奨金対象となる。報奨金金額については Meta の裁量による（具体的金額は不明）。 ● 届け出た脆弱性に対してポイントが加算され、ダイヤモンドからブロンズまで 5 ランク付けがされ、そのランクにより、標準報奨金アワードの支払いに乘算されるボーナスや特別特典などが加わる。 URL: https://www.facebook.com/whitehat/hackerplus/

表 4-16 バグバウンティプログラム例 (Google LLC)

企業	Google LLC
名称	Bug Hunters
報 償 内 容など	● Android: Pixel Titan M (セキュリティチップ) の脆弱性について最大 100 万ドル ● Chrome: サンドボックスの脆弱性について最大 30 万ドル ● Google and Alphabet Vulnerability Reward Program (VRP): 最大\$31,337 (実績ベース) ● Google Play: 最大 2 万ドル ● Open-source Security 補助金: (影響力のある OSS に対して) 最大 3 万ドル URL: https://bughunters.google.com/

表 4-17 バグバウンティプログラム例 (サイボウズ株式会社)

企業	サイボウズ株式会社
名称	サイボウズ脆弱性報奨金制度
報 償 内 容など	● 脆弱性最大 100 万円 ● 安全に検証を行うため「脆弱性検証環境提供プログラム」を用意 ● 一部の参加者に対し、通常の報奨金制度とは異なる条件や限定製品への検証を許可するプライベートプログラムを用意 URL: https://cybozu.co.jp/products/bug-bounty/

表 4-18 バグバウンティプログラム例 (LINE 株式会社)

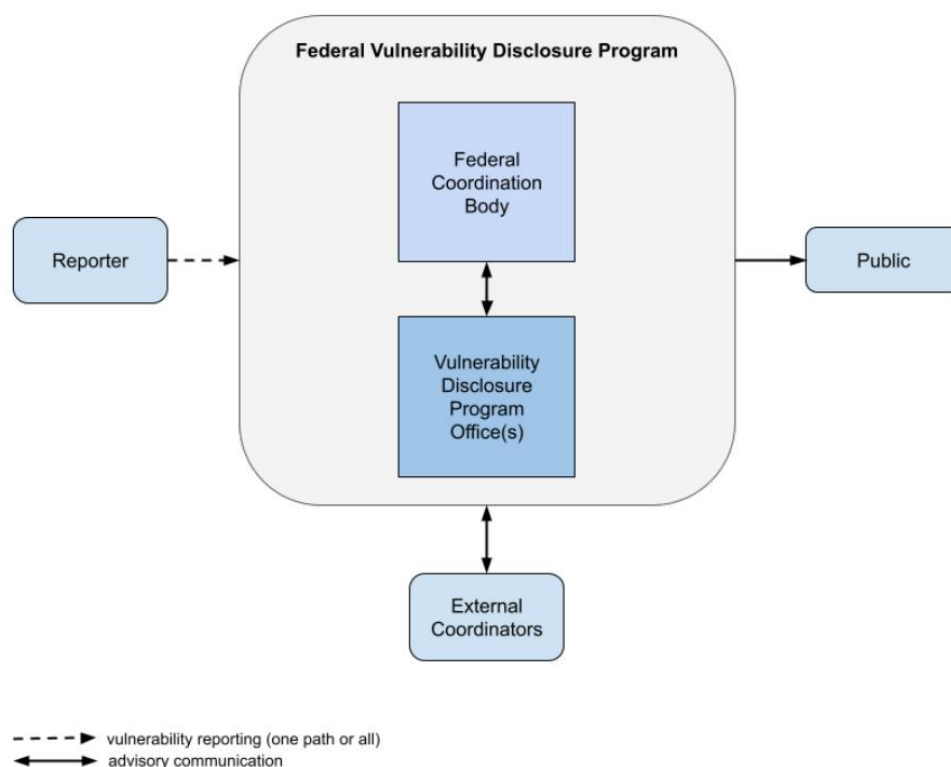
企業	LINE 株式会社
名称	LINE Security Bug Bounty Program
報 償 内 容など	脆弱性報告のプラットフォームは Hackerone を利用 ● Remote Code Execution (RCE): 最大 3 万ドル ● Server Side Request Forgery (SSRF): 最大 2 万ドル ● SQL Injection: 最大 2 万ドル URL: https://bugbounty.linecorp.com/ja/

4.1.5.2.2. バグバウンティプログラムと脆弱性情報流通

バグバウンティプログラムによって脆弱性情報が収集される以前は、外部の第三者による脆弱性の報告は発見者が開発者に対して善意の通知を行うか、あるいは CSIRT に報告を送り、そこが脆弱性情報のコーディネーションを行って発見者と開発者との調整を行う。発見者が直接開発者に対して通知を送る場合、発見者と開発者の間でコミュニケーションに齟齬が発生しトラブルが発生する可能性がある。脆弱性情報を扱う第三者組織が発見者（報

告者）と開発者を仲介し脆弱性対応をコーディネーションするニーズが生まれ CSIRT が介在するようになった。その後、脆弱性情報のハンドリングが公的な意味を持つようになり、現在では CVE といった世界的な脆弱性管理フレームワークが作られている。脆弱性情報が脆弱性ハンドリング組織（主に各国の CSIRT、日本では JPCERT/CC と IPA）に報告された後、脆弱性ハンドリング組織が開発者に通知ができない場合、あるいは開発者が脆弱性の対応を行わないと脆弱性ハンドリング組織が判断した場合、一定の期間（おおよそ 45 日が目安とされる）が過ぎると脆弱性ハンドリング組織は脆弱性について公開することができる。脆弱性情報のハンドリングは、当初は発見者と開発者の仲介というものであったが、現在では公的な組織としての意味合いを含むようになった。これまで脆弱性情報のコーディネーションを行う組織は、脆弱性対応のプロセスにおいて、（企業からの自社製品に関する脆弱性以外では）第三者である善意の脆弱性発見者・報告者を暗黙の前提としていた。

現在、米国政府では、連邦脆弱性報告ガイドライン（Federal Vulnerability Disclosure Guidelines/NIST SP 800-216 Draft¹⁴²）を策定中である。その中では、次のような構造を提案している。これは 2020 年に制定された The Internet of Things Cybersecurity Improvement Act of 2020 をうけての対応である。どこまでの範囲で脆弱性流通のコントロールをするのかは未知数であるが、脆弱性流通のコントロールというのはさらに公的な位置づけになること可能性が高い。



出所）NISC Federal Vulnerability Disclosure Guidelines/NIST SP 800-216 Draft

図 4-8 米国政府脆弱性情報管理の概略（案）

¹⁴² SP 800-216 (Draft) Recommendations for Federal Vulnerability Disclosure Guidelines
<https://csrc.nist.gov/publications/detail/sp/800-216/draft>

一方で、現在、大手の企業では独自のバグバウンティプログラムを立ち上げて、脆弱性情報を積極的に収集（購入）している。また、脆弱性を発見するためのプラットフォームを用意して提供する形で、脆弱性情報の収集を推進している。このようなビジネス的側面を持った脆弱性流通の枠組みも拡大していくことが十分に予想できる。

このように公的な意味における脆弱性流通の拡大及び強化と、民間における脆弱性流通（収集）の拡大と強化が同時に進んでいるのが分かる状況にある。このような、これまでとは大きく変化しつつある中、OSS の脆弱性流通もこれまでとは枠組みが拡大・変化していく可能性は高い。

4.2. OSS を活用におけるセキュリティリスクへの有効性の高い対応策

2 章、3 章及び 4.1 節の調査結果も踏まえ、3 章に示した各観点（OSS の評価や選定、保守、ライセンス管理・構成管理、脆弱性管理）について OSS を活用におけるセキュリティリスクへの有効性の高い対応策を以下に示す。

- **開発者の開発環境による構成管理（ライセンス管理・構成管理）**

2 章、3 章に示した構成管理では、ビルドシステム等の OSS に備わっている機能を活用しながら、ソフトウェアの依存関係を管理している事例や検証ツールの自動化による早期検証や省力化を実現した事例もあった。

- **OSS の脆弱性対策自動監査（脆弱性管理）**

4.1 節の OSS 活用における課題及び脆弱性情報等の共有及び共有体制では、脆弱性への対策の課題があった。脆弱性情報を収集し、利用している OSS に対する影響度を確認し、影響度が高い場合は、速やかに対策を実施すること（利用停止や異なる OSS への移行も含む）が重要となる。

- **長期利用に関するリスク（OSS の保守）**

長期的な OSS 活用におけるリスク（脆弱性管理やアップデート対応や対応に関するコスト等）の考え方を顧客と事前に合意する等の事例があった。

- **OSS 活用開始前の審査や確認（OSS の評価や選定、ライセンス管理）**

2 章、3 章では、OSS の取り扱いに関するガバナンスが取れており、利用してよい OSS が組織内で予めリストアップされ、複数の審査を経たうえで OSS の利用が可能となっている事例。また、開発部と OSS 管理部門のリーダー同士がリスクの確認や懸念点を議論する会議で決定する事例。さらに、開発プロジェクトにおけるリスクに対する第三者チェック機能がある事例があった。

5. SBOM を導入や活用するサプライチェーンモデルの構築に向けた調査・実証

5.1. 目的

サプライチェーンを通じた SBOM の活用・普及が進まない要因や課題の特定や、その阻害要因の解消や課題解決に向けて、実際のサプライチェーンにおける実証を通じた評価・検討を行う。特に、普及・促進の主な阻害要因と考えられるコストと効果に関して実際のソフトウェアに対する評価を行うことで、課題解決に向けた方向性について整理する。

5.2. SBOM 導入・活用に関する課題・障害等の整理

効果的な実証に向けて要件を定義するため、SBOM の導入・活用に関する課題・障害等に関する観点を整理する。本調査及び昨年度までの調査に基づき整理すると、下記のとおり、導入時の障害等と、導入後の障害等に分類することができる。導入時の課題・障害等は、導入に至らない原因に関わるものであり、導入後の課題・障害等は、導入済みの組織が抱える効率化に関わる問題である。

表 5-1 SBOM 導入・活用に関する主な課題・障害

区分	No	障害・課題等（情報源）
（導入意思決定・インセンティブ等） 導入時の障害等	1-1	費用対効果が明確ではない。採算が取れる適用範囲が明確では無い。（第 5 回タスクフォース等）
	1-2	中小サプライヤーなどにとって SBOM の有効性自体が理解されず、協力が得られない。（ヒアリング）
	1-3	取引企業間で責任の明確化、コスト分担についてのコンセンサスが形成されていない（ヒアリング）
	1-4	政府、業界における基準化や発注者による SBOM の要件化が進んでいない。（第 5 回タスクフォース等）
	1-5	サプライヤーの知財権の懸念から SBOM が提供されない。（事例集、ヒアリング）
	1-6	商用ツールは価格が高額である。（ヒアリング）
（部品管理の効率化） 導入後の障害等	2-1	ツールの機能や性能が成熟しておらず効率的に管理できない（第 4 回タスクフォース）
	2-2	ツール活用やプラクティスにおける効率的なノウハウが整備されていない（第 5 回タスクフォース等）
	2-3	SBOM フォーマットや項目の標準化、共通化が浸透していない。（第 5 回タスクフォース等）
	2-4	開発者であるサプライヤーからの SBOM 提供が得られず、調達者による部品特定は非効率（ヒアリング）
	2-5	再帰的な部品まで厳格に管理すると大きなコスト負担が生じる（ヒアリング）

これらの障害・課題を踏まえて、次節以降で SBOM 実証の要件検討・設計を行う。

5.3. SBOM 実証の設計・計画

5.3.1. SBOM 実証に関する要件

SBOM 実証においては、SBOM の導入・活用の阻害要因や課題を解決するために有用な知見を得ることが重要である。実際のソフトウェアとサプライチェーンにおける SBOM 適用を通じて、5.2 に挙げた障害・課題等に対してできるだけ多くの項目を評価・検討できるように、本実証では以下のようなことを要件として設定し、それに基づき SBOM 実証の設計・実施を行う。

- **SBOM の有効性の評価（課題・障害 1-2）**

中小サプライヤーなど SBOM や部品管理自体を導入していない企業は多い。そのような企業のうち、SBOM の有効性自体に理解が得られていない企業にとっては、SBOM の有効性が示されなければ、SBOM 導入のインセンティブは働かない。そのため、実際のソフトウェアを対象に SBOM の有効性を示すことが必要である。

- **SBOM のコスト・効果に関する計測・評価（課題・障害 1-1, 1-6, 2-2, 2-3, 2-4, 2-5）**

SBOM の有効性・必要性を認識する企業であっても、コストと効果の具体的な情報が把握できず、SBOM 導入の意思決定ができないケースもある。また、サプライチェーンを通じて再帰的に利用される部品を特定し、SBOM を作成するコストが大きく、SBOM による部品管理をどのように、どの程度実施することが現実的かつ効果的かも明確ではない。実証においては、SBOM の実施項目や比較シナリオを整理し、それらの項目に関わるコストと効果の計測・評価を行うことで、SBOM が効果的となる適用方法や条件を評価することが必要である。

- **サプライチェーンにおける取引企業を含む評価（課題・障害 1-3）**

サプライチェーンを通じたソフトウェア開発が拡大する中で、サプライチェーンを通じた企業間の関係におけるコスト・効果の評価や課題抽出が期待される。

- **その他、SBOM に関わる障害・課題の抽出**

SBOM 導入・活用を促進するためには、実践を通じて課題・障害を抽出し、それを解消することが重要である。SBOM の実証を通じて障害・課題を抽出し、課題やノウハウの整理を行うことが期待される。

5.3.2. SBOM の有効性の評価の考え方

SBOM の有効性を評価するにあたって留意すべき点がある。SBOM の必要性については、製品・サービスの最終ベンダーや大手企業においてはおおむね理解は得られつつある。最終ベンダーにとっては、利用者に対して一次責任を負う立場にあり、利用者がセキュリティ事故を起こした場合、直接的な対応を迫られるためである。一方で、中小サプライヤーにとっては SBOM や部品管理の必要性について認知されているとは必ずしも言えない。なお、本調査では、SBOM による部品管理と、個社の独自形式による部品管理を区別する。SBOM に

よる部品管理は、取引企業間で部品情報の共有を前提とし、機械処理可能な標準フォーマットによる部品管理とする。従来、Excel などを用いて独自形式による OSS の部品管理を行う企業は存在するが、標準化されないフォーマットでは、企業間の部品情報の共有や自動化に大きな障害が生じるためである。

これまで部品管理自体を実施していない企業にとっては、SBOM 形式、独自形式によらず新たに部品管理を行う場合、部品管理コスト自体は確実に増加するため、コストだけを見れば SBOM の導入によりコストは増大する。SBOM の有効性を評価するためには、3 段階の比較の考え方が必要である。

1 段階目の評価は、部品管理なしと SBOM による部品管理の比較である（下図①）。この場合、SBOM による部品管理は確実にコストが増大するが、脆弱性リスクの低減効果が得られるため、部品管理コストとリスク低減効果の比較が必要になる。ところが、リスク低減効果は、確率的な事象によるもので金額換算が困難であるため、部品管理コストの増大との費用対効果分析は難しい。

2 段階目の評価は、部品管理を実施することは前提とし、独自形式による部品管理と SBOM 形式による部品管理の比較である（下図②）。この比較を行うためには、部品管理の範囲・レベルを揃えた比較が必要である。独自形式の部品管理の対象部品は少なく、SBOM 形式の部品管理の対象部品が多ければ、公平な比較による SBOM の有効性の評価ができないためである。

さらに 3 段階目の評価は、SBOM の適用レベルに応じた費用対効果の評価による SBOM の有効性評価である。OSS 部品の再帰的な部品まで厳格に SBOM 管理を行おうとすれば、SBOM ツールでは誤検知・検出漏れがあることが分かっているため、人手による膨大な精査コストが発生し、SBOM 部品管理の採算性が取れなくなるケースがある。このケースの評価は、OSS の事例に依存し、コストが大きく異なるため、一般化した評価は難しいと考えられる。

以上のように、SBOM の有効性評価は上記 3 段階の評価の観点に分けて評価が考えられる。そのうち、本実証においては、予算規模と実証期間も考慮して、2 段階目の評価を行うこととし、独自形式による部品管理と SBOM 形式の部品管理の範囲・レベルを同一条件として比較評価するものとする。

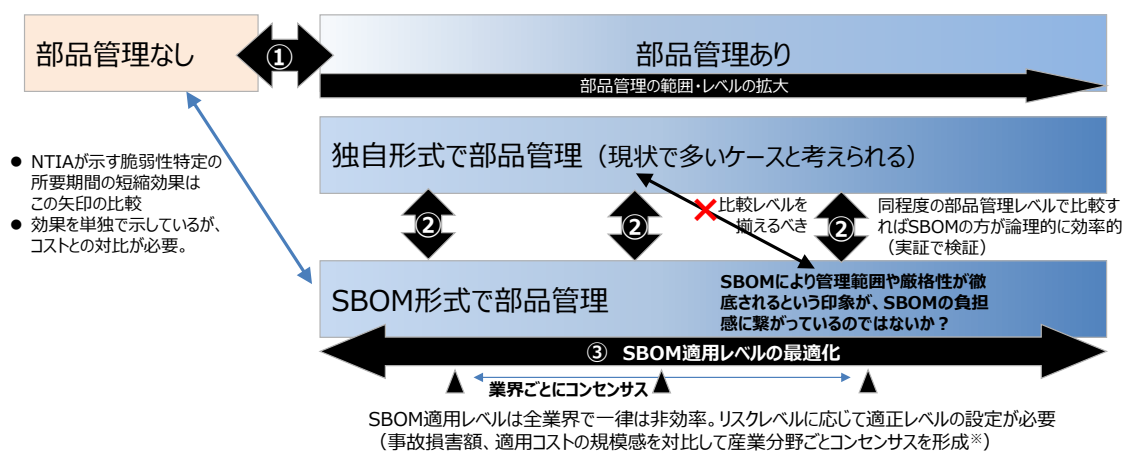


図 5-1 SBOM の有効性に関する比較評価の考え方の整理

5.3.3. SBOM 実証の対象と体制の検討

SBOM 実証の対象分野、ソフトウェア、企業を選定するうえで以下の観点が重要である。

- **SBOM の重要性・優先度が高い分野**

SBOM 検討で先行する米国 NTIA における PoC では、自動車分野、医療機器分野、ソフトウェア分野が優先項目として重要分野・優先分野が挙げられている。これらの分野は、脆弱性等によるリスクが高い分野又は政府による基準化が進む分野であり優先度が高い。日本国内においても同様の理由から優先度は高い。

- **サプライチェーンを含む評価**

SBOM はユーザー企業を含むサプライチェーンを通じた部品情報の共有を前提として検討されているものであり、サプライチェーンを通じた部品管理が必要な場合に効果を発揮するものである。したがって、実証においてもサプライチェーンを含む取引企業間での評価が重要である。

以上のような観点及び前節の SBOM 実証の要件を満たすことができる実証事例を対象とする企業群の探索、選定、打診を行った結果、以下のような対象と体制で実施することが決定した。

表 5-2 対象分野及びソフトウェア

対象分野	自動車分野（開発検証ツール）
対象ソフトウェア名称	GARDEN Scenario Platform
製品ベンダー	株式会社 NTT データ オートモビリジェンス研究所 (NTT データ ARC)
ソフトウェア概要	• 自動運転システム開発向け検証基盤ソフトウェア。 • 自動運転ソフトウェアの安全性評価のための機能動作シミュレーションのシナリオ生成機能を提供。 • モデリング、走行データ分類、軌跡抽出道路編集、シナリオ組合せテスト、シナリオ実行 • オープンソースとして、ソースコードを公開。

実証におけるサプライチェーンは以下のとおり、ソフトウェアサプライヤー、製品ベンダー、製品のユーザー企業から構成される。

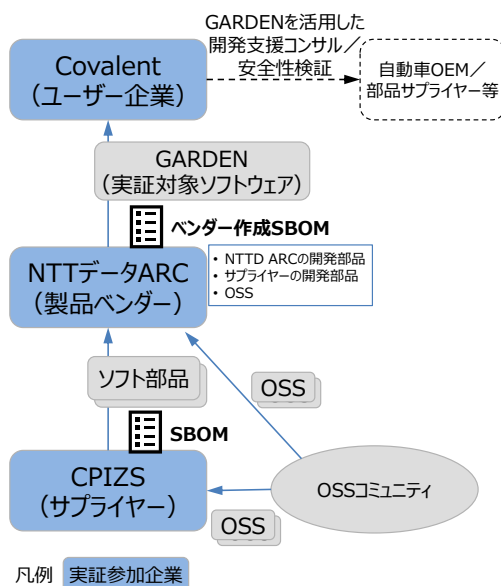


図 5-2 実証におけるソフトウェア・サプライチェーンの構成

5.3.4. SBOM 実証の構成と設計

5.3.4.1. 実証の基本構成

ソフトウェアタスクフォースにおける議論に基づき実証の基本構成は以下のとおりとした。比較項目として、SBOM の効果・コストの基本要素を対象とし、比較条件としての主要要素の組合せを基に、比較評価シナリオを導出し、コスト・効果の計測・評価、SBOM の効果的な活用方法の検討や整理を行うこととした。

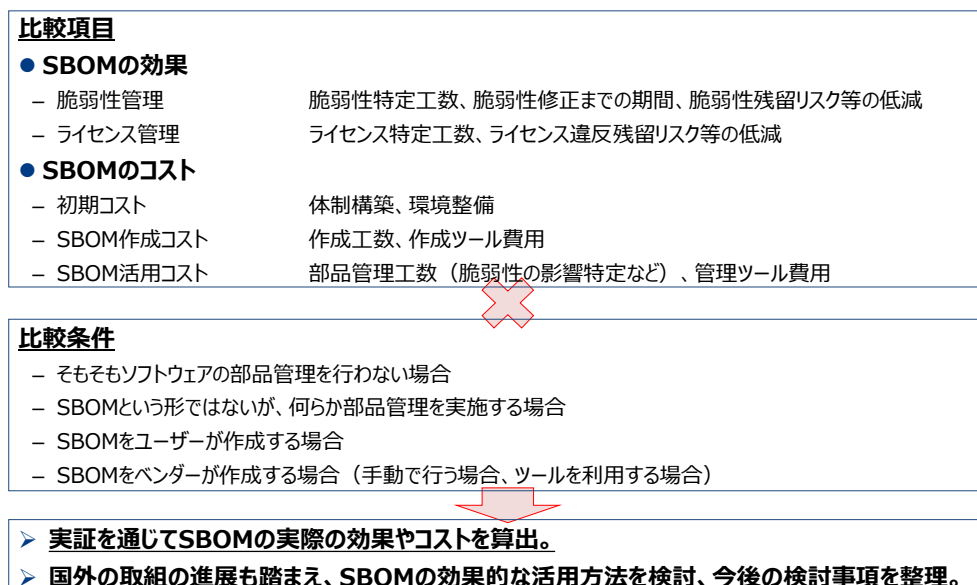


図 5-3 実証項目の基本構成の考え方

5.3.4.2. 実証項目の設計

前節の考え方にに基づき実証項目の設計を示す。評価項目は、コストと効果に二分される。コストは初期導入コストと運用コストに分かれ、前者は、ツール環境整備、要員の学習などからなり、後者は、主に **SBOM** 作成、活用からなる。効果には、大きく分けて脆弱性管理、ライセンス管理があり、そのうち脆弱性管理の効果としては、主に脆弱性対応期間の短縮、脆弱性残留リスクの低減、脆弱性特定コストの低減が挙げられる。

一方、比較条件に関しては、大きく以下の 4 ケースに分けられる。

[1] **SBOM** 無し

従来企業独自に実施している部品管理による方法。本実証では、実証の予算や機関の制約からサプライチェーンにおける開発者が直接利用する部品（「部品情報 1 階層のみ」と呼ぶ）に限定する。

[2] **SBOM** 有り（手動生成）

開発企業が直接利用する部品について **SBOM** を手動で生成し、サプライチェーンを通じて共有し、脆弱性特定などツールを用いて **SBOM** を活用する。

[3] **SBOM** 有り（自動生成）無償ツール

開発企業がツールを用いて **SBOM** を自動生成し、サプライチェーンを通じて共有し、脆弱性特定などツールを用いて **SBOM** を活用する。

[4] **SBOM** 有り（自動生成）有償ツール

ユーザー企業が **SBOM** 有償ツールを用いて、**SBOM** の生成・活用を行う。

以上の評価項目×比較条件の組合せにより比較評価シナリオを設定し、実証を通じてコスト・効果の計測・評価を行う。

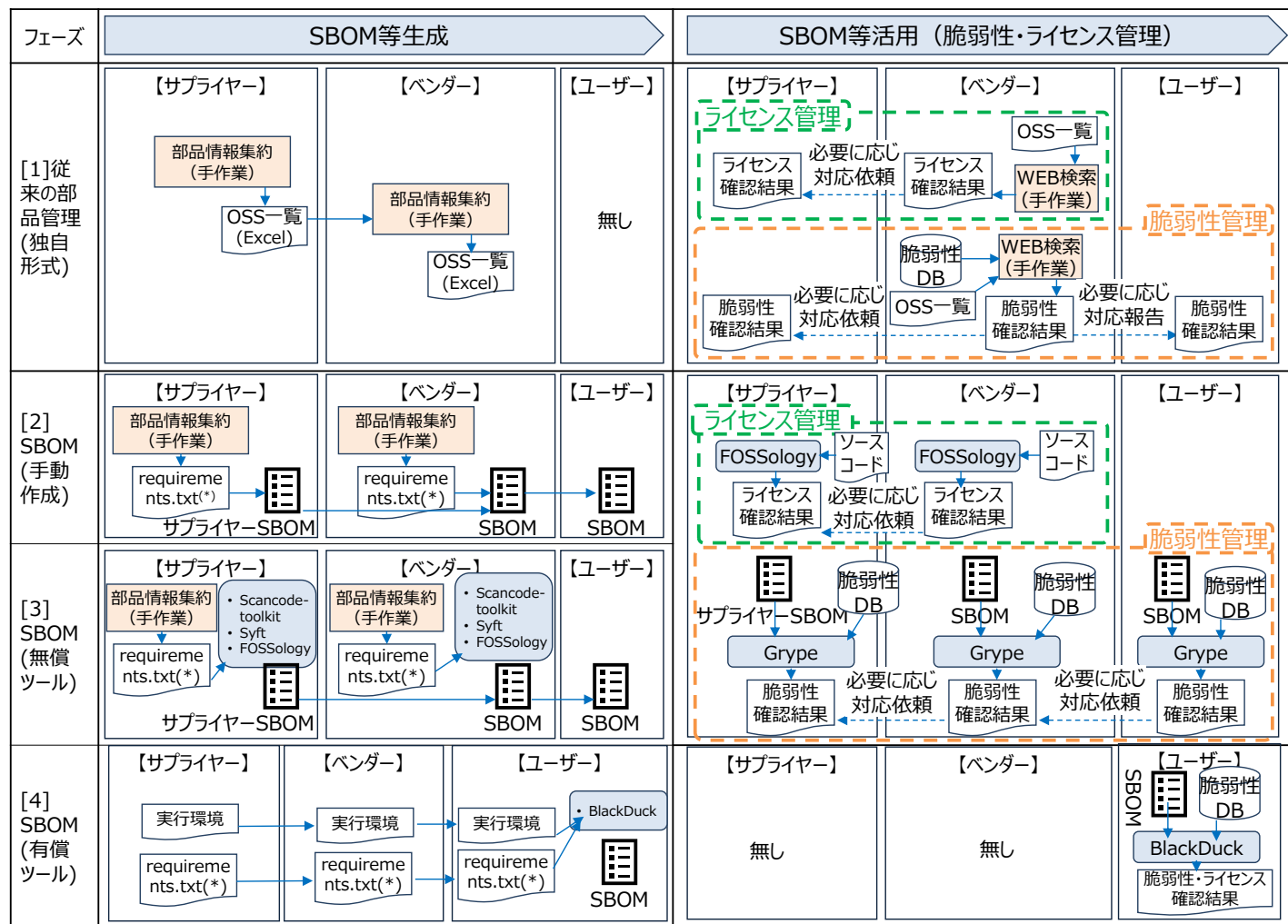
表 5-3 SBOM 実証の評価項目及び比較評価方法に関する検討

			比較検証観点	[1] SBOM 無し（独自形式の部品情報 1 階層のみ）	[2] SBOM 有り（手動生成）	[3] SBOM 有り（無償ツール）	[4] SBOM 有り（有償ツール）	
コスト	初期導入	体制構築・環境整備		SBOM 有無に関わらず部品管理をする場合、一定のコストが発生する。	SBOM 形式ではない部品管理の準備工数を評価する（SBOM 習得が不要なため工数は少ない。）	開発者が、ビルドツール構成管理情報などを用いて、SBOM を手動で生成するための準備・環境構築等の工数を評価する。	ツール利用法など体制準備の工数を評価。	ツール初期費用、ツール利用法など体制準備の工数等を評価
	運用	SBOM 作成	作成工数	開発者は構成管理情報を使うことで効率的に生成できる。ツール生成と誤検知の修正工数合計と開発者が手動生成する工数コストを比較する。	開発者が直接利用する部品情報 1 階層を独自形式で作成する工数を評価する。	開発者が自身のソフトの構成管理情報を用いて部品情報を特定し SBOM 形式にする工数を評価する。構成管理情報を活用することで、効率化できることを示す。	SBOM ツールで生成する工数を評価する。[2]と比較する。	SBOM ツールで生成する工数を評価する。[2]と比較する。
			作成ツール費用	無償ツールにより、有償ツールの利用料を低減できる。（SBOM 精度はツールに依存するが、精度は効果として評価する。）	発生しない	発生しない	無償ツールの費用は 0 とする。	有償ツールの使用料などを確認する。
		SBOM 活用	部品管理工数(脆弱性、ライセンス)	SBOM 無しの場合、手動で脆弱性、ライセンスを特定する必要がある、SBOM による自動の場合と工数を比較する。	部品情報から手動で (1)NVD 等の脆弱性 DB の検索、 (2)ライセンス情報の特定する工数を評価する。	SBOM から自動特定すると想定する。	SBOM ツールから自動特定すると想定する。	SBOM ツールから自動特定すると想定する。
			活用ツール費用	無償ツールがあれば、有償ツールに比べコストを低減できる。	発生しない	発生しない	無償ツールの費用は 0 とする。	有償ツールの使用料などを確認する。

			比較検証観点	[1] SBOM 無し（独自形式の部品情報 1 階層のみ）	[2] SBOM 有り（手動生成）	[3] SBOM 有り（無償ツール）	[4] SBOM 有り（有償ツール）
効果	脆弱性管理	脆弱性対応期間の短縮	SBOM 有りと SBOM 無し（独自形式）について脆弱性の特定、修正の所要期間を試算する。SBOM 無しの場合、間接的な利用部品の脆弱性を認識する状況と修正までの期間を試算する。	部品情報 1 階層と仮定し、部品の認識、修正にかかる状況（シナリオ）と期間を試算する。	SBOM 有りの場合、すべての部品の脆弱性認識は即時と仮定し、修正は自社で可能な場合と供給者が修正を必要とする場合に分けて期間を試算する。	[2]と同じ	[2]と同じ
		脆弱性の残留リスクの低減	開発者により手動で作成される SBOM の場合、部品情報が正確と仮定し、SBOM 無し（部品情報 1 階層）と比較し、脆弱性の検出漏れのリスクが低減できる程度を示す。	部品情報 1 階層のみと仮定し、[2] SBOM 有り（手動）で全階層のほぼ正確な部品情報が得られた場合と比較し、脆弱性の検出漏れの可能性を例示し、脆弱性残留の比率を推計する。	構成管理情報を用いて SBOM を生成している場合、部品・脆弱性の特定は正確と仮定する。	開発者が SBOM を供給する場合[2]と同様に正確と仮定する。	開発者が SBOM を供給する場合[2]と同様に正確と仮定する。
		脆弱性特定コストの低減	SBOM により脆弱性特定を自動化することで工数低減の程度を示す。	標準化されていないソフト部品名などから NVD などの脆弱性 DB を手動検索し、脆弱性情報を特定する工数を評価する。	SBOM 有りの場合、脆弱性特定は自動化できるという前提に基づき、コスト（脆弱性特定ツールの費用）などを評価する。	脆弱性特定は自動化できると仮定する。[2]と同じ。	脆弱性特定は自動化できると仮定する。[2]と同じ。
	ライセンス	ライセンス違反特定コストの低減	SBOM を用いてライセンス特定を自動化することにより、削減できる工数を示す。	部品情報から手動でライセンス情報を取得する工数コストを評価する。SBOM 有り・自動でライセンス特定する場合とコスト比較する。	SBOM 有りの場合、ツールで自動化する場合のコスト（ライセンス特定ツールの費用）を評価する。	ライセンス違反の特定は自動化できると仮定する。[2]と同じ。	ライセンス違反の特定は自動化できると仮定する。[2]と同じ。

			比較検証観点	[1] SBOM 無し（独自形式の部品情報 1 階層のみ）	[2] SBOM 有り（手動生成）	[3] SBOM 有り（無償ツール）	[4] SBOM 有り（有償ツール）
	管理	ライセンス違反のリスク低減	開発者が作成する SBOM は正確と仮定し、ライセンス違反の検出漏れリスクが低減できることを示したい。	（脆弱性の残留リスク低減と同様の評価）部部品情報 1 階層のみと仮定し、[2] SBOM 有りで全階層の正確な部品情報が得られた場合と比較し、ライセンスの検出漏れの可能性を例示し、脆弱性残留の比率を試算する。	SBOM 有りの場合、正確に部品のライセンスが特定できると仮定する。	開発者が SBOM を供給する場合[2]と同様に正確に特定できると仮定する。	開発者が SBOM を供給する場合[2]と同様に正確に特定できると仮定する。

以上のような評価の考え方にに基づき、実証の評価シナリオを具体的に示したものが下図である。下図においては、サプライチェーンの主体（サプライヤー、ベンダー、ユーザー）ごとに、SBOM 生成・活用において実施する作業について、入力とする情報、利用するツール、出力結果などについて具体的に示している。



(*)ソフトウェアインストールに必要なOSSモジュールが記載。SW自体のデバッグ（インストール可否確認）を通じて記載漏れが検知されるため、部品の記載漏れは開発過程で防止される。

図 5-4 実証における比較評価シナリオ

5.3.5. SBOM 実証の実施スケジュール

下記のとおり、実証の要件定義・設計の後に、実証事業者との連携により実証を進め、計測・評価・分析を経て、結果の取りまとめを行った。

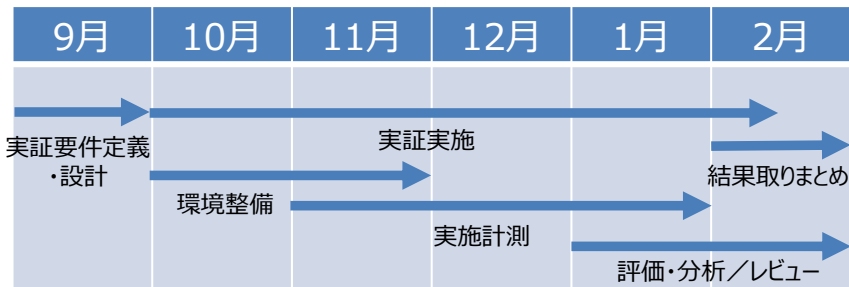


図 5-5 実証の実施スケジュール

5.4. SBOM 実証の実施結果

5.4.1. 実証の手順

5.4.1.1. コストの測定

SBOM の比較評価項目に応じてコストの測定を行うため、以下の手順で実証を推進した。

表 5-4 実証におけるコスト測定手順

コスト費目	シナリオ	実証の手順
初期導入時の体制構築・環境整備コスト	[1] SBOM 無し（独自形式の部品情報 1 階層のみ）の場合	部品・使用 OSS に関する情報集約環境整備の工数を調査した。具体的には、開発者（サプライヤー、製品ベンダー）にて、目安工数の調査を実施した。
	[2] SBOM 有り（手動生成）の場合	SBOM 作成フォーマットと作成要領を作成した。開発者にて、SBOM 活用ツール（脆弱性検出ツール等）が読込可能な SBOM の形式を調査し、手動で作成する SBOM の要件を特定した。また、SBOM の手動作成の方法を調査し、SBOM 作成要領を習得した。 本シナリオにおいても、脆弱性・ライセンス検出は自動化ツールにより行われるため、「[3] SBOM 有り（自動）の場合」で測定される、OSS 脆弱性管理・OSS ライセンス管理ツールの環境整備工数を算入した。現実的な SBOM 運用を考慮し、OSS 脆弱性管理ツールはユーザー・製品ベンダー・サプライヤーすべて、OSS ライセンス管理ツールは開発者において導入・運用されることとして工数を計測している。
	[3] SBOM 有り（自動）の場合	SBOM 生成ツール・OSS 脆弱性管理・OSS ライセンス管理ツールにつきインストール方法を調査し実施のうえ、動作確認を実施した。これらの作業の一連の工数を測定した。 一般的に企業におけるツール導入時には、ツールの操作方法などを記した作業手順書が作成されることを踏まえ、実際に作業手順書を作成し、作業内容習得に使用可能かレビュー後、修正を加えるまでの一連の工数を測定した。 現実的な SBOM 運用を考慮し、SBOM 生成ツールは製品ベンダーとサプライヤー、OSS 脆

コスト費目	シナリオ	実証の手順
		弱性管理ツールはユーザー・製品ベンダー・サプライヤーすべて、OSS ライセンス管理ツールは開発者において導入・運用されることとして工数を計測している。 なお、今回の実証では SBOM 生成ツールとして、Linux Foundation の SPDX Workgroup がサポートする Scancode-Toolkit と、脆弱性検出ツールとして使用した Grype と同一の開発元により開発され連動性が担保された Syft の 2 種類を使用した。 OSS 脆弱性管理ツールには、Linux Foundation の SPDX Workgroup がサポートするものが無かったため、SBOM 生成ツールとして使用した Syft と同一の開発元により開発され連動性が担保された Grype を使用した。 OSS ライセンス管理ツールには、Linux Foundation の SPDX Workgroup がサポートする FOSSology を使用した。
	[4] SBOM 有り（自動・有償ツール）の場合	有償の SBOM 生成ツール・OSS 脆弱性管理・OSS ライセンス管理ツールにつきインストール方法を調査し実施のうえ、動作確認を実施した。これらの作業の一連の工数を測定した。 ツールとして、米国の Synopsys が開発・販売している“Black Duck”を採用した。 具体的には、公開情報、開発元からのマニュアルを読み、スキャンによる SBOM 作成と OSS 部品の誤検知精査・修正、OSS 部品の脆弱性特定、OSS 部品のライセンス特定を実施するための操作方法を習得するまでの作業工数を測定した。なお、ツールの操作方法などを記した作業手順書の作成は、開発元からのマニュアルにより代替可能なため割愛した。 操作方法の習得においては、有償ツールによる SBOM の共有可能性を検証するため、SBOM を出力して他者へ共有することの可否確認・他者から受け取った SBOM の読み込みの可否及び対応フォーマットの確認も実施した。ただし、これらの確認は[1][2][3]においては実施していないため、工数の計測・計上は実施していない。

コスト費目	シナリオ	実証の手順
		一般的に企業における有償ツール導入時には、開発元・販売元との営業担当者との折衝が発生する。そこで、当該工数も初期導入時の体制構築・環境整備コストに計上している。 有償ツールは導入にあたり費用が発生するため、導入時の初期費用の調査を実施した。ただし、本シナリオで実際に試用したツールの初期費用はツール開発元の機密情報に該当するため、本報告書では調査結果の掲載、及び実証結果への反映は割愛した。 現実的な SBOM 運用を考慮し、有償ツールはユーザーにおいて導入・運用されることとして工数を計測している。有償ツールは、バイナリデータからの SBOM 生成が可能なものもあるため、ソースコードの提供を受けられないユーザー企業での採用可能性が高いと考えられるためである。
運用時のコスト (SBOM 作成コスト、作成工数の計測)	[1] SBOM 無し (独自形式の部品情報 1 階層のみ) の場合	開発者にて、使用された OSS 部品に関する情報を収集し、Excel の OSS 一覧に取りまとめる作業の目安工数を調査。
	[2] SBOM 有り (手動生成) の場合	SBOM を手作業で作成し、その工数を計測した。開発者の開発部分の SBOM は、開発者が作成した。サプライヤー開発部分の SBOM は、製品ベンダーが作成した場合と、サプライヤーが作成した場合を想定して、工数に差異が発生するか見積を行った。
	[3] SBOM 有り (自動) の場合	SBOM 生成ツールを用いて SBOM を生成した。OSS の誤検出・検出漏れのレビューのため、SBOM 生成ツールが SBOM 生成時に読み込みを行ったファイルを特定のうえ、当該ファイルから確認可能な OSS と SBOM 上に記載された OSS を比較した。これらの作業の一連の工数を測定した。 現実的な SBOM 運用を考慮し、SBOM 生成ツールは製品ベンダーとサプライヤーで導入・運用されることとして工数を計測している。
	[4] SBOM 有り (自動・有償ツール) の場合	有償ツールを用いて、[3]と同様の実証作業を実施した。ただし、本シナリオでは有償ツールはユーザーにおいて導入・運用されることとして工数を計測している。

コスト費目	シナリオ	実証の手順
運用時のコスト (SBOM 作成コスト、作成ツール費用の調査)	[1] SBOM 無し(独自形式の部品情報 1 階層のみ) の場合	SBOM 作成ツールを使用しないため、実証作業は実施せず。
	[2] SBOM 有り(手動生成) の場合	SBOM 作成ツールを使用しないため、実証作業は実施せず。
	[3] SBOM 有り(自動) の場合	有償の SBOM 生成ツールの初期導入・ランニングコストについて公開情報、ツール開発元・販売元ヒアリングにより調査した。
	[4] SBOM 有り(自動・有償ツール) の場合	今回の実証で使用した有償ツールの導入時の費用の調査を実施した。ただし、本シナリオで実際に試用したツールの費用はツール開発元の機密情報に該当するため、本報告書では調査結果の掲載、及び実証結果への反映は割愛した。
運用時のコスト (SBOM 活用コスト、部品管理工数(脆弱性管理、ライセンス管理) の計測)	[1] SBOM 無し(独自形式の部品情報 1 階層のみ) の場合	脆弱性の特定のため、製品ベンダーにて、GARDEN Scenario Analyzer における一部の OSS 部品について脆弱性データベース(NVD・JVN) で脆弱性を検索し、工数を測定した。 OSS 部品のライセンスの特定のため、製品ベンダーにて、GARDEN Scenario Analyzer における一部の OSS 部品について Web 検索によるライセンス情報の調査を実施し、工数を測定した。
	[2] SBOM 有り(手動生成) の場合	SBOM から自動特定するため、当初は工数が発生しないものと仮定したが、脆弱性管理ツール・ライセンス管理ツールの操作に工数が発生することが判明し、その際の工数を測定した。
	[3] SBOM 有り(自動) の場合	SBOM から自動特定するため、当初は工数が発生しないものと仮定したが、脆弱性管理ツール・ライセンス管理ツールの操作に工数が発生することが判明し、その際の工数を測定した。
	[4] SBOM 有り(自動・有償ツール) の場合	有償ツールを用いて、[3]と同様の実証作業を実施した。
運用時のコスト (SBOM 活用コスト、活用ツール費用の調査)	[1] SBOM 無し(独自形式の部品情報 1 階層のみ) の場合	SBOM 活用ツールを使用しないため、実証作業は実施せず。
	[2] SBOM 有り(手動	[3]の結果を援用するため、実証作業は実施せ

コスト費目	シナリオ	実証の手順
	生成) の場合	ず。
	[3] SBOM 有り (自動) の場合	有償の OSS 脆弱性・ライセンス検出ツールの初期導入・ランニングコストについて公開情報、ツール開発元・販売元ヒアリングにより調査した。
	[4] SBOM 有り (自動・有償ツール) の場合	今回の実証で使用した有償ツールの導入時の費用の調査を実施した。ただし、本シナリオで実際に試用したツールの費用はツール開発元の機密情報に該当するため、本報告書では調査結果の掲載、及び実証結果への反映は割愛した。

5.4.1.2. 効果の測定

SBOM の比較評価項目に応じて効果の測定を行うため、以下の手順で実証を推進した。

表 5-5 実証における効果測定手順

コスト費目	シナリオ	実証の手順
脆弱性管理における効果の測定 (脆弱性対応期間の短縮効果の測定)	[1] SBOM 無し (独自形式の部品情報 1 階層のみ) の場合	製品ベンダー・サプライヤーにて、部品の脆弱性特定・部品の脆弱性修正に必要な作業の目安工数の見積を実施した。
	[2] SBOM 有り (手動生成) の場合	製品ベンダー・サプライヤー・ユーザーにて、SBOM を用いて OSS 部品の脆弱性を特定し、特定された脆弱性を修正する工数の見積を実施した。
	[3] SBOM 有り (自動) の場合	製品ベンダー・サプライヤー・ユーザーにて、SBOM を用いて OSS 部品の脆弱性を特定し、特定された脆弱性を修正する工数の見積を実施した。
	[4] SBOM 有り (自動・有償ツール) の場合	ユーザーにて、SBOM を用いて OSS 部品の脆弱性を特定し、特定された脆弱性の対処 (利用の中断等) を検討する工数の見積を実施した。
脆弱性管理における効果の測定 (脆弱性の残留リスク低減効果の測定)	[1] SBOM 無し (独自形式の部品情報 1 階層のみ) の場合	独自形式の SBOM を使用し検出された脆弱性が、製品ベンダーにおいて特定漏れとなっていないかを確認した。
	[2] SBOM 有り (手動生成) の場合	構成管理情報を用いて SBOM を生成している場合、部品・脆弱性の特定はほぼ完全と仮定するため、実証作業は実施せず。

コスト費目	シナリオ	実証の手順
	[3] SBOM 有り（自動）の場合	製品ベンダーが SBOM を供給する場合[2]と同様にほぼ正確に特定できると仮定するため、実証作業は実施せず。
	[4] SBOM 有り（自動・有償ツール）の場合	有償ツールを用いて、SBOM から脆弱性を特定し、誤検知（対応不要・対応済みの脆弱性）事例が著しく多くないか、検知漏れ（製品ベンダー・サプライヤーが認識している脆弱性の未特定）事例が著しく多くないか確認することで、「部品・脆弱性の特定ほぼ完全と仮定」とするのが正しいか検証した。
脆弱性管理における効果の測定 （脆弱性特定コストの低減効果の測定）	[1] SBOM 無し（独自形式の部品情報 1 階層のみ）の場合	同シナリオにおける「部品管理工数（脆弱性管理、ライセンス管理）の計測」の結果を援用するため、実証作業は実施せず。
	[2] SBOM 有り（手動生成）の場合	コスト（OSS 脆弱性管理ツールの費用）などを評価した。無償ツールの動作確認が取れば、ツール使用に伴うコストは発生しないものと評価した。脆弱性管理ツールの操作に工数が発生することが判明し、その際の工数を測定した。
	[3] SBOM 有り（自動）の場合	[2]の結果を援用するため、実証作業は実施せず。
	[4] SBOM 有り（自動・有償ツール）の場合	ユーザーにて、SBOM を用いて OSS 部品の脆弱性を特定する工数を測定した。
ライセンス管理における効果の測定 （ライセンス違反特定コストの低減効果の測定）	[1] SBOM 無し（独自形式の部品情報 1 階層のみ）の場合	部品管理工数（脆弱性管理、ライセンス管理）の計測のため、実証作業は実施せず。
	[2] SBOM 有り（手動生成）の場合	コスト（OSS 脆弱性管理ツールの費用）などを評価。無償ツールの動作確認が取れば、ツール使用に伴うコストは発生しないものと評価した。ライセンス管理ツールの操作に工数が発生することが判明し、その際の工数を測定した。
	[3] SBOM 有り（自動）	[2]の結果を援用するため、実証作業

コスト費目	シナリオ	実証の手順
ライセンス管理における 効果の測定 (ライセンス違反のリスク 低減効果の測定)	の場合	は実施せず。
	[4] SBOM 有り（自動・ 有償ツール）の場合	ユーザーにて、SBOM を用いて OSS 部品のライセンスを特定する工数を 測定した。
	[1] SBOM 無し（独自形 式の部品情報 1 階層の み）の場合	SBOM を使用し検出されたライセン スが、製品ベンダーにおいて特定漏れ となっていないかを確認した。
	[2] SBOM 有り（手動生 成）の場合	本実証では、SBOM が正確と仮定し、 作業は実施せず。
	[3] SBOM 有り（自動） の場合	本実証では、SBOM が正確と仮定し、 作業は実施せず。
	[4] SBOM 有り（自動・ 有償ツール）の場合	有償ツールを用いて、SBOM からライ センスを特定し、誤検知（製品ベンダ ー・サプライヤーが認識していないラ イセンスの表示）事例が著しく多くな いか、検知漏れ（製品ベンダー・サブ ライヤーが認識しているライセンス の未特定）事例が著しく多くないか を、を確認することで、「ほぼ正確に 部品のライセンスが特定」できるかを 検証した。

5.4.2. 独自形式による部品管理

5.4.2.1. コスト評価（独自形式）

5.4.2.1.1. 初期コスト（独自形式）

初期導入時の体制構築・環境整備コストとして、工数が合計 1.83 人時間発生した。

製品ベンダーに 1.83 人時間の工数が発生しており、サプライヤー及びユーザーの工数は発生していない。

上記の工数には、環境整備の工数と体制構築・学習の工数が含まれる。それぞれの主な業務内容は以下のとおりである。

表 5-6 初期コストの主な内容（独自形式）

初期コストの項目	主な業務内容説明
環境整備	OSS 一覧フォーマットを作成し、社内レビューのうえ修正した。
体制構築・教育	OSS 一覧の運用に必要な資料を作成し、社内レビューのうえ修正した。 サプライヤーへの OSS 一覧フォーマット・運用を連携した。

環境整備には合計 1.33 人時間が発生した。製品ベンダーに 1.33 人時間が発生しており、サプライヤー、ユーザーには工数は発生していない。体制構築・学習には合計 0.5 人時間が発生した。製品ベンダーに 0.5 人時間が発生しており、サプライヤー、ユーザーには工数は発生していない。

5.4.2.1.2. 運用コスト（独自形式）

5.4.2.1.2.1. 部品情報生成・共有コスト

各項目において発生したコストは下記のとおり。

- SBOM 作成工数

- 独自形式の部品情報作成時のコストとして、工数が合計 0.79 人時間発生した。製品ベンダーに 0.44 人時間、サプライヤーに 0.35 人時間の工数が発生しており、ユーザーの工数は発生していない。
上記の工数には、作成の工数と精査の工数が含まれる。それぞれの主な業務内容は以下のとおりである。

表 5-7 部品情報生成・共有コストの主な内容（独自形式）

部品情報生成・共有コストの項目	主な業務内容説明
部品情報作成	OSS 一覧へ、使用した OSS 部品を起票した。
部品情報精査	起票された OSS 一覧をレビューした。具体的には、記載漏れ・誤記載等が無いか、フォーマットとおりに記載されているかを確認した。

作成には合計 0.35 時間が発生した。サプライヤーに 0.35 人時間が発生しており、製品ベンダー、ユーザーの工数は発生していない。精査には合計 0.44 人時間が発生した。製品ベンダーに 0.44 人時間が発生しており、サプライヤー、ユーザーの工数は発生していない。

- 作成ツール費用

- 独自形式の部品情報作成にツールは使用されておらず、作成ツール費用は発生していない。

5.4.2.1.2.2. 脆弱性管理コスト

- 脆弱性対応期間

ソフトウェア・アップデートが四半期に 1 回行われるため、脆弱性発見から製品ベンダーが脆弱性の存在を確実に把握するまでに 3 か月要する。脆弱性が発見された際に、すべての使用 OSS 部品の脆弱性が点検するために 3-5 日間要するため、脆弱性の発見から修正までのリードタイム目安は 3 か月+3-5 日間と見積もられる。
これは、その際にと仮定した場合のリードタイム目安である。

- 脆弱性の残留リスク

SBOM 有りの場合に脆弱性が検出されなかったため、独自形式の部品情報を使用時の製品ベンダーによる脆弱性の検知漏れは確認されなかった。

- 脆弱性特定コスト

- 脆弱性特定コストとして、工数が合計 0.79 人時間発生した。製品ベンダーに 0.44 人時間、サプライヤーに 0.35 人時間の工数が発生しており、ユーザーの工数は発生していない。

5.4.2.1.2.3. ライセンス管理コスト

- ライセンス特定コスト

- ライセンス特定コストとして、工数が合計 1.17 人時間発生した。サプライヤーに 1.17 人時間が発生しており、製品ベンダー、ユーザーの工数は発生していない。

5.4.3. SBOM による部品管理

5.4.3.1. コスト評価（SBOM 手動）

5.4.3.1.1. 初期コスト（SBOM 手動）

工数が合計 150.32 人時間発生した。サプライヤーに 62.72 人時間、製品ベンダーに 62.72 人時間、ユーザーに 24.88 人時間の工数が発生した。

上記の工数には、環境整備の工数と体制構築・学習の工数が含まれる。それぞれの主な業務内容は以下のとおりである。

表 5-8 初期コストの主な内容（SBOM 手動）

初期コストの項目	主な業務内容説明
環境整備	脆弱性特定ツール・ライセンス特定ツールにつき、インストールを手順調査のうえ実施した。SBOM 活用ツールで使用可能な SBOM 作成の手順を把握するため、脆弱性検出ツールの処理を調査し、SBOM 手動作成を試行しながら作成方法を確認した。
体制構築・教育	脆弱性特定ツール・ライセンス特定ツールにつき、操作方法の調査を実施し作業手順書を作成した。SBOM 作成方法を把握可能な SBOM テンプレートを整備した。 上記の作業手順書、SBOM テンプレートを内部レビューした。 SBOM の手動作成方法を、作業手順書・SBOM テンプレート、及び SBOM 手動作成・脆弱性検出ツールへの読込試行を通じて理解した。

環境整備には合計 102.09 人時間が発生した。サプライヤーに 41.92 人時間、製品ベンダーに 41.92 人時間、ユーザーに 18.25 人時間が発生している。体制構築・学習には合計 48.23 人時間が発生した。サプライヤーに 20.80 人時間、製品ベンダーに 20.80 人時間、ユーザー

に 6.63 人時間が発生している。

5.4.3.1.2. 運用コスト（SBOM 手動）

5.4.3.1.2.1. 部品情報生成コスト（SBOM 手動）

- SBOM 作成工数
 - SBOM 作成時のコストとして、工数が合計 0.33 人時間発生した。サプライヤーに 0.33 人時間の工数が発生しており、製品ベンダー、ユーザーの工数は発生していない。
上記の工数には、作成の工数が含まれる。作成には合計 0.33 時間が発生した。サプライヤーに 0.33 人時間が発生しており、製品ベンダー、ユーザーの工数は発生していない。精査には工数が発生していない。
- 作成ツール費用
 - SBOM 生成ツールを使用していないため、作成ツール費用は発生していない。

5.4.3.1.2.2. 脆弱性管理コスト（SBOM 手動）

- 脆弱性対応期間

SBOM 有りの場合、脆弱性検出ツールを用いて日々脆弱性を確認することで、脆弱性発生から製品ベンダーが脆弱性の存在を把握するまでのリードタイムは削減できる。脆弱性を修正するまでのリードタイムのみが発生するため、脆弱性対応期間の目安は 3-5 日間である。
- 脆弱性の残留リスク

手動生成した SBOM を脆弱性検出ツール Grype に入力した結果、脆弱性は検出されなかった。
- 脆弱性特定コスト

脆弱性特定コストとして、工数が合計 0.05 人時間発生した。サプライヤーに 0.02 人時間、製品ベンダーに 0.02 人時間、ユーザーに 0.02 人時間が発生した。

5.4.3.1.2.3. ライセンス管理コスト（SBOM 手動）

- ライセンス特定コスト
 - ライセンス特定コストとして、工数が合計 0.03 人時間発生した。サプライヤーに 0.02 人時間、製品ベンダーに 0.02 人時間が発生しており、ユーザーの工数は発生していない。なお、有償ツールの適用結果から、無償ツールでは、部品検出漏れがあることが確認されたが、それらは、SBOM 生成の精査コスト又は脆弱性リスクに影響するものであり、本実証では比較条件の留意点として挙げるにとどめ、ライセンス管理コストには含めなかった。

5.4.3.2. コスト評価（SBOM 自動）

5.4.3.2.1. 初期コスト（SBOM 自動）

工数が合計 128.25 人時間発生した。サプライヤーに 51.68 人時間、製品ベンダーに 51.68 人時間、ユーザーに 24.88 人時間が発生している。

上記の工数には、環境整備の工数と体制構築・学習の工数が含まれる。

表 5-9 初期コストの主な内容（SBOM 自動）

初期コストの項目	主な業務内容説明
環境整備	SBOM 生成ツール・脆弱性特定ツール・ライセンス特定ツールにつき、インストールを手順調査のうえ実施した。
体制構築・教育	SBOM 生成ツール・脆弱性特定ツール・ライセンス特定ツールにつき、操作方法の調査を実施し作業手順書を作成した。 上記の作業手順書を内部レビューした。具体的には、作業手順書のとおりツールのインストール・操作が可能かを確認した。

環境整備には合計 71.08 人時間が発生した。サプライヤーに 26.42 人時間、製品ベンダーに 26.42 人時間、ユーザーに 18.25 人時間が発生している。体制構築・学習には合計 51.17 人時間が発生した。サプライヤーに 25.27 人時間、製品ベンダーに 25.27 人時間、ユーザーに 6.63 人時間が発生している。

なお、環境構築にあたり、複数の無償ツールのインストールを試行した。その結果発生した障害の内容と対応結果は下記のとおり。

表 5-10 無償ツールの試行において発生した障害・対応結果・得られたノウハウ

ツール	障害	対応結果	ノウハウ
Scancode-Toolkit	Windows 環境でのインストールを試行時、 ・Windows Subsystem for Linux や、Ubuntu のインストールが必要と判明（開発元文書には明記なく、未整備だった） ・開発元文書の指示に則り、最新版の Python（バージョン 3.10）をインストールした結果、インストーラに Python が認識されずエラー発生	・Windows Subsystem for Linux や、Ubuntu のインストールを実施 ・Python のバージョンを、開発元文書でサポートする旨明記されている旧バージョン（バージョン 3.9）に変更 上記対策実施結果、インストール成功。SBOM 生成の動作確認済み。	動作環境を確認してから必要なソフトをインストールするのが効率的
Syft	インストール時に特段の障害は発生せず。		

ツール	障害	対応結果	ノウハウ
SPDX SBOM Generator	・Linux 環境にインストール実施し、動作確認結果、SBOM 生成のために、Linux 内への環境の追加が必要な他 SW があることが判明。	インストールを断念。	
Sourceclear	・インストール方法が、開発元から公開されていない。 ・その他の公開情報から調査実施も、インストール方法を明示した情報が確認されない。	インストールを断念。	
Vuls	・ツール動作確認結果、開発中のソースコードを読み込んで脆弱性検出ができないことが判明。（サーバー管理用途で、サーバー内の OS・アプリケーションをスキャン・脆弱性検出する機能のみ実装されている）	インストールを断念。	
Grype	・Windows 非対応	・Linux 環境（VirtualBox Ubuntu）整備のうえ、インストールを実施	動作環境を確認してから必要なソフトをインストールするのが効率的
FOSSology	・Web ベースでの動作を確認		

5.4.3.2.2. 運用コスト（SBOM 自動）

5.4.3.2.2.1. 部品情報生成コスト（SBOM 自動）

● SBOM 作成工数

SBOM 作成時のコストとして、工数が合計 0.37 人時間発生した。製品ベンダーに 0.18 人時間、サプライヤーに 0.18 人時間が発生しており、ユーザーの工数は発生していない。

上記の工数には、作成の工数と精査の工数が含まれる。作成には合計 0.03 時間が発生した。サプライヤーに 0.02 人時間、製品ベンダーに 0.02 人時間が発生しており、ユーザーの工数は発生していない。精査には合計 0.34 人時間が発生した。サプライヤーに 0.17 人時間、製品ベンダーに 0.17 人時間が発生しており、

ユーザーの工数は発生していない。

なお、Syft・FOSSology では SPDX 準拠の SBOM が生成可能な機能が確認された。そこで、当該 SBOM において、SPDX 規格における” Package Information” の全 23 項目が反映されているか確認を実施した。結果、Syft では 6 項目、FOSSology では 10 項目の生成が確認された。各項目の確認結果は下表のとおり。

表 5-11 SPDX 項目に対する Syft 生成 SBOM での対応状況

SPDX 項目	Syft 生成 SBOM (SPDX 形式) での対応状況	
	有無	記載例
7.1 Package name field	無	
7.2 Package SPDX identifier field	有	<u>"SPDXID": "SPDXRef-Package-python-SPARQLWrapper-1.8.5",</u> "name": "SPARQLWrapper", "licenseConcluded": "NONE", "downloadLocation": "NOASSERTION", "externalRefs": [
7.3 Package version field	無	
7.4 Package file name field	無	
7.5 Package supplier field	無	
7.6 Package originator field	無	
7.7 Package download location field	無	
7.8 Files analyzed field	有	<i>"filesAnalyzed": false,</i> "licenseDeclared": "NONE", "sourceInfo": "acquired package info from installed python package manifest file: work/Zipc_airflow/src/requirements.txt", "versionInfo": "1.8.5"
7.9 Package verification code field	無	
7.10 Package checksum field	無	
7.11 Package home page field	無	
7.12 Source information field	有	"filesAnalyzed": false, "licenseDeclared": "NONE", <i>"sourceInfo": "acquired package info from installed python package manifest file: work/Zipc_airflow/src/requirements.txt",</i> "versionInfo": "1.8.5"
7.13 Concluded license field	有	"SPDXID": "SPDXRef-Package-python- SPARQLWrapper-1.8.5", "name": "SPARQLWrapper", <i>"licenseConcluded": "NONE",</i> "downloadLocation": "NOASSERTION",

SPDX 項目	Syft 生成 SBOM (SPDX 形式) での対応状況	
	有無	記載例
		"externalRefs": [
7.14 All licenses information from files field	無	
7.15 Declared license field	有	"filesAnalyzed": false, "licenseDeclared": "NONE", "sourceInfo": "acquired package info from installed python package manifest file: work/Zipc_airflow/src/requirements.txt", "versionInfo": "1.8.5" },
7.16 Comments on license field	無	
7.17 Copyright text field	無	
7.18 Package summary description field	無	
7.19 Package detailed description field	無	
7.20 Package comment field	無	
7.21 External reference field	有	"SPDXID": "SPDXRef-Package-python-SPARQLWrapper-1.8.5", "name": "SPARQLWrapper", "licenseConcluded": "NONE", "downloadLocation": "NOASSERTION", "externalRefs": [{ "referenceCategory": "SECURITY", "referenceLocator": "cpe:2.3:a:python-SPARQLWrapper:python-SPARQLWrapper:1.8.5:*:*:*:*:*" , "referenceType": "cpe23Type" }, ...
7.22 External reference comment field	無	
7.23 Package attribution text field	無	

SPDX 項目	FOSSology 生成 SBOM (SPDX 形式) での対応状況	
	有無	記載例
7.1 Package name field	無	
7.2 Package SPDX identifier field	無	
7.3 Package version field	無	

SPDX 項目	FOSSology 生成 SBOM (SPDX 形式) での対応状況	
	有無	記載例
7.4 Package file name field	有	<pre> <spdx:Package rdf:about="http://dc9869de0243/repo/SPDX2_Zi pc_airflow.zip_1641702981-spx.rdf#SPDXRef- upload2"> <spdx:name>Zipc_airflow.zip</spdx:name> <spdx:packageFileName>Zipc_airflow.zip</spdx:packageFileName> </pre>
7.5 Package supplier field	無	
7.6 Package originator field	無	
7.7 Package download location field	有	<pre> <spdx:Package rdf:about="http://dc9869de0243/repo/SPDX2_Zi pc_airflow.zip_1641702981-spx.rdf#SPDXRef- upload2"> <spdx:name>Zipc_airflow.zip</spdx:name> <spdx:packageFileName>Zipc_airflow.zip</spdx:packageFileName> <spdx:downloadLocation rdf:resource="http://spdx.org/rdf/terms#noassertion" /> </pre>
7.8 Files analyzed field	無	
7.9 Package verification code field	有	<pre> <spdx:Package rdf:about="http://dc9869de0243/repo/SPDX2_Zi pc_airflow.zip_1641702981-spx.rdf#SPDXRef- upload2"> <spdx:name>Zipc_airflow.zip</spdx:name> <spdx:packageFileName>Zipc_airflow.zip</spdx:packageFileName> <spdx:downloadLocation rdf:resource="http://spdx.org/rdf/terms#noassertion"/> <spdx:packageVerificationCode> <spdx:PackageVerificationCode> <spdx:packageVerificationCodeValue>104658e 9d848404175acee2125e70a9f734c2706</spdx:packageVerificationCodeValue> </spdx:PackageVerificationCode> </spdx:packageVerificationCode> </pre>
7.10 Package checksum field	有	<pre> <spdx:checksum> <spdx:Checksum> <spdx:algorithm rdf:resource="http://spdx.org/rdf/terms#checksumAlgorithm_sha1"/> <spdx:checksumValue>bd5dd4a7e3c935860061 273dad428bf138aec0bc</spdx:checksumValue> </pre>

SPDX 項目	FOSSology 生成 SBOM (SPDX 形式) での対応状況	
	有無	記載例
		<code></spdx:Checksum></code> <code></spdx:checksum></code>
7.11 Package home page field	無	
7.12 Source information field	無	
7.13 Concluded license field	有	<code><spdx:licenseConcluded</code> <code>rdf:resource="http://spdx.org/rdf/terms#noas</code> <code>sertion" /></code>
7.14 All licenses information from files field	有	<code><spdx:licenseInfoFromFiles</code> <code>rdf:resource="http://spdx.org/rdf/terms#noas</code> <code>sertion" /></code>
7.15 Declared license field	有	<code><spdx:licenseInfoFromFiles</code> <code>rdf:resource="http://spdx.org/rdf/terms#noas</code> <code>sertion" /></code>
7.16 Comments on license field	有	<code><spdx:licenseComments><![CDATA[</code> <code>licenseInfoInFile determined by</code> <code>Scanners:</code> <code>- nomos ("unknown".b04657)</code> <code>- monk ("unknown".b04657)</code> <code>- ojo ("unknown".b04657)</code> <code>- reso (unknown.b04657)</code> <code>]]></spdx:licenseComments></code>
7.17 Copyright text field	有	<code><spdx:copyrightText</code> <code>rdf:resource="http://spdx.org/rdf/terms#noas</code> <code>sertion" /></code>
7.18 Package summary description field	無	
7.19 Package detailed description field	無	
7.20 Package comment field	有	<code><rdfs:comment></code> <code>This document was created using license</code> <code>information and a generator from FOSSology.</code> <code></rdfs:comment></code>
7.21 External reference field	無	
7.22 External reference comment field	無	
7.23 Package attribution text field	無	

Scancode-Toolkit・Syft から出力された JSON 形式の SBOM は、SPDX 規格への準拠は確認されなかった。また、その他の SBOM の国際規格である SWID・CycloneDX への準拠も確認されなかった。開発元から、各項目の意味内容の定義及び国際規格との対応関係は公開されていない。開発元への問い合わせも実施したが、対応関係を把握可能な情報は開発元から

提供されなかった。したがって、国際規格との対応関係の確認には至らなかった。

本実証においては、各ツールで生成された SBOM につき、大統領令の最小要件への対応関係の確認を実施した。ただし大統領令自体には、最小要件の各情報項目への説明が記載されているのみで、最小要件の各情報項目と実際の SBOM の情報項目との対応関係は明確に規定されていない。そこで、NTIA が別途公表している国際規格と最小要件の対応関係、大統領令の説明から推察される対応関係を基に、本実証で生成された SBOM の情報項目と最小要件の対応関係を整理した。

NTIA は、大統領令の最小要件で定められた情報項目と SPDX・SWID・CycloneDX の情報項目との対応関係につき見解を公表している¹⁴³。NTIA の見解に基づき、Syft・FOSSology が生成した SPDX 準拠の SBOM における大統領令の最小要件への対応状況を確認した結果は下表のとおり。

表 5-12 実証で作成した SPDX 準拠の SBOM における SPDX 項目の有無

SPDX 項目（主な項目）		実証で作成した SPDX 準拠の SBOM における有無	
		Syft	FOSSology
DCI	6.8 Creator	△ ※ SBOM 作成ツール 情報／ツールベンダー 情報が記載	×
	6.9 Created	○	○
Package Information	7.1 Package name field	×	×
	7.2 Package SPDX identifier field	○	×
	7.3 Package version field	×	×
	7.4 Package file name field	×	○
	7.5 Package supplier field	×	×

Scancode-Toolkit・Syft から出力された JSON 形式の SBOM は、国際規格への準拠は確認されなかったため、NTIA 見解を基にした大統領令の最小要件への対応関係整理には至らなかった。

大統領令の説明と各ツールで生成された SBOM の情報の内容から、大統領令の最小要件の項目と対応関係があると推察される SBOM の情報項目は下表のとおり。

表 5-13 各ツールの SBOM における対応項目名

	各ツールの SBOM における対応項目名		
	ScanCode-Toolkit (JSON 形式)	Syft (JSON 形式)	Syft (SPDX 形式)
コンポーネントのサ プライヤー名	-	-	-

¹⁴³ Survey of Existing SBOM Formats and Standards (2021) Table 1: A mapping between SPDX, CycloneDX, and SWID https://www.ntia.gov/files/ntia/publications/sbom_formats_survey-version-2021.pdf

	各ツールの SBOM における対応項目名		
	ScanCode-Toolkit (JSON 形式)	Syft (JSON 形式)	Syft (SPDX 形式)
コンポーネント名	-	"name"	"name"
コンポーネントのバージョン	"requirement"	"version"	"versionInfo"
その他の一意の識別子	"purl" ※Package URL を記載	"cpes" ※CPE を記載 "purl" ※Package URL を記載	"referenceLocator" ※CPE を記載
依存関係	-	-	-
SBOM データの作成者	-	-	-
タイムスタンプ	"start_timestamp"	-	"created"

Scancode-Toolkit・Syft から出力された JSON 形式の SBOM について、開発元へ大統領令の最小要件との対応関係につき問い合わせも実施したが、対応関係を把握可能な情報は開発元から提供されなかった。

5.4.3.2.2.2. 脆弱性管理コスト（SBOM 自動）

- 脆弱性対応期間
SBOM 有りの場合、脆弱性検出ツールを用いて日々確認することで、脆弱性発生から製品ベンダーが脆弱性の存在を把握するまでのリードタイムは削減できる。脆弱性を修正するまでのリードタイムのみが発生するため、脆弱性対応期間の目安は 3-5 日間である。
- 脆弱性の残留リスク
Syft により生成された SBOM を脆弱性検出ツール Gype に入力した結果、脆弱性は検出されなかった。
なお、Scancode-Toolkit により生成された SBOM、及び Syft により生成された SPDX 形式の SBOM は、Gype により読込成功しなかった。これは、Gype が脆弱性特定のために必要とする CPE が SBOM 内に含まれていないことが原因と考えられる。
- 脆弱性特定コスト
脆弱性管理のコストとして、工数が合計 0.05 人時間発生した。ユーザーに 0.02 人時間、製品ベンダーに 0.02 人時間、サプライヤーに 0.02 人時間が発生している。

5.4.3.2.2.3. ライセンス管理コスト（SBOM 自動）

- ライセンス特定コスト
 - ライセンス特定コストとして、工数が合計 0.03 人時間発生した。サプライヤーに 0.02 人時間、製品ベンダーに 0.02 人時間が発生しており、ユーザーの工数は発生していない。なお、有償ツールの適用結果から、無償ツールでは、部品検出

漏れがあることが確認されたが、それらは、SBOM 生成の精査コスト又は脆弱性リスクに影響するものであり、本実証では比較条件の留意点として挙げるにとどめ、ライセンス管理コストには含めなかった。

5.4.3.3. コスト評価（SBOM 自動・有償ツール）

5.4.3.3.1. 初期コスト（SBOM 自動・有償ツール）

工数が合計 9.5 人時間発生した。ユーザーに 9.5 人時間が発生しており、サプライヤー、製品ベンダーの工数は発生していない。

上記の工数には、環境整備の工数と体制構築・学習の工数が含まれる。環境整備には合計 3.0 時間が発生した。ユーザーに 3.0 人時間が発生しており、サプライヤー、製品ベンダーの工数は発生していない。体制構築・学習には合計 6.5 時間が発生した。ユーザーに 6.5 人時間が発生しており、サプライヤー、製品ベンダーの工数は発生していない。

5.4.3.3.2. 運用コスト（SBOM 自動・有償ツール）

SBOM 作成時のコストとして、工数が合計 9.87 人時間発生した。製品ベンダーに 5.83 人時間、ユーザーに 4.03 人時間が発生しており、サプライヤーの工数は発生していない。

上記の工数には、作成の工数と精査の工数が含まれる。作成には合計 0.03 人時間が発生した。ユーザーに 0.03 人時間が発生しており、サプライヤー、製品ベンダーの工数は発生していない。精査には合計 9.83 人時間が発生した。製品ベンダーに 5.83 人時間、ユーザーに 4.00 人時間が発生しており、サプライヤーの工数は発生していない。

製品ベンダーの工数は、新たに検出された OSS 部品の精査・ユーザー対応に 5.83 人時間が発生し、検知結果確認での工数は発生していない。ユーザーの工数は、検知結果確認が 0.17 人時間、新たに検出された OSS 部品の精査・製品ベンダー確認が 3.83 人時間発生している。

なお、Black Duck では SPDX 準拠の SBOM が生成可能な機能が確認された。そこで、当該 SBOM において、SPDX 規格における”Package Information”の全 23 項目が反映されているか確認を実施した。各項目の確認結果は下表のとおり。

表 5-14 Black Duck 生成 SBOM での SPDX 対応状況

SPDX 項目	Black Duck 生成 SBOM での対応状況	
	有無	記載例
7.1 Package name field	有	<pre> "filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the </pre>

SPDX 項目	Black Duck 生成 SBOM での対応状況	
	有無	記載例
		externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" },
7.2 Package SPDX identifier field	有	{ "SPDXID": "SPDXRef-Package-influxdb-python-531", "annotations": [], "copyrightText": "Copyright (c) 2020 InfluxDB", "description": "NOASSERTION", "downloadLocation": "NOASSERTION", "externalRefs": [
7.3 Package version field	有	"filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" },
7.4 Package file name field	有	"filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" },

SPDX 項目	Black Duck 生成 SBOM での対応状況	
	有無	記載例
7.5 Package supplier field	有	<pre> "filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "<i>packageSupplier</i>": "<i>Organization: pypi</i>", "versionInfo": "5.3.1" }, </pre>
7.6 Package originator field	無	
7.7 Package download location field	有	<pre> "SPDXID": "SPDXRef-Package-influxdb-python-531", "annotations": [], "copyrightText": "Copyright (c) 2020 InfluxDB", "description": "NOASSERTION", "<i>downloadLocation</i>": "<i>NOASSERTION</i>", "externalRefs": [</pre>
7.8 Files analyzed field	有	<pre> <i>filesAnalyzed</i>: <i>false</i>, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" }, </pre>
7.9 Package verification code field	無	

SPDX 項目	Black Duck 生成 SBOM での対応状況	
	有無	記載例
7.10 Package checksum field	無	
7.11 Package home page field	有	<pre> "filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" }, </pre>
7.12 Source informatio n field	無	
7.13 Concluded license field	有	<pre> "filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" }, </pre>
7.14 All licenses informatio n from files field	無	

SPDX 項目	Black Duck 生成 SBOM での対応状況	
	有無	記載例
7.15 Declared license field	有	<pre> "filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" }, </pre>
7.16 Comments on license field	有	<pre> "filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" }, </pre>
7.17 Copyright text field	有	<pre> "SPDXID": "SPDXRef-Package-influxdb-python-531", "annotations": [], "copyrightText": "Copyright (c) 2020 InfluxDB", "description": "NOASSERTION", "downloadLocation": "NOASSERTION", "externalRefs": [</pre>
7.18 Package summary description field	無	

SPDX 項目	Black Duck 生成 SBOM での対応状況	
	有無	記載例
7.19 Package detailed description field	有	<pre>"SPDXID": "SPDXRef-Package-influxdb-python-531", "annotations": [], "copyrightText": "Copyright (c) 2020 InfluxDB", "description": "NOASSERTION", "downloadLocation": "NOASSERTION", "externalRefs": [</pre>
7.20 Package comment field	有	<pre>"filesAnalyzed": false, "licenseComments": "The concluded license was taken from the package level", "licenseConcluded": "MIT", "licenseDeclared": "MIT", "name": "influxdb-python", "packageComment": "This is an open source component from the Black Duck Knowledge Base which was automatically detected as a directly declared dependency, the PackageSupplier was based on the externalNamespace", "packageFileName": "NOASSERTION", "packageHomepage": "NOASSERTION", "packageSupplier": "Organization: pypi", "versionInfo": "5.3.1" },</pre>
7.21 External reference field	有	<pre>"externalRefs": [{ "referenceCategory": "PACKAGE_MANAGER", "referenceLocator": "pkg:pypi/influxdb@5.3.1", "referenceType": "purl" }, { "referenceCategory": "OTHER", "referenceLocator": "https://poc101.blackduck.synopsys.com/api/components/d935b03f-f0af-4b7d-8278-68611148d47a", "referenceType": "BlackDuckHub-Component" }, { "referenceCategory": "OTHER", "referenceLocator": "https://poc101.blackduck.synopsys.com/api/components/d935b03f-f0af-4b7d-8278-68611148d47a/versions/b447aa26-4ebe-4b02-b2bd-7bea32670d24", "referenceType": "BlackDuckHub-Component-Version" }],</pre>

SPDX 項目	Black Duck 生成 SBOM での対応状況	
	有無	記載例
7.22 External reference comment field	無	
7.23 Package attribution text field	無	

本実証においては、各ツールで生成された SBOM につき、大統領令の最小要件への対応関係の確認を実施した。NTIA は、大統領令の最小要件で定められた情報項目と SPDX・SWID・CycloneDX の情報項目との対応関係につき見解を公表している。NTIA の見解に基づき、Black Duck が生成した SPDX 準拠の SBOM における大統領令の最小要件への対応状況を確認した結果は下表のとおり。

表 5-15 Black Duck が生成した SBOM における大統領令最小要件の対応状況¹⁴⁴

SPDX 項目（主な項目）		実証で作成した SPDX 準拠の SBOM における有無
		Black Duck
DCI	6.8 Creator	△※ ※ SBOM 作成ツール情報／ツールベンダー情報が記載
	6.9 Created	○
Package Information	7.1 Package name field	○
	7.2 Package SPDX identifier field	○
	7.3 Package version field	○
	7.4 Package file name field	○
	7.5 Package supplier field	○

5.4.3.3.2.1. 脆弱性管理コスト（SBOM 自動・有償ツール）

工数が合計 0.02 人時間発生した。ユーザーに 0.02 人時間が発生しており、サプライヤー、製品ベンダーの工数は発生していない。

5.4.3.3.2.2. ライセンス管理コスト（SBOM 自動・有償ツール）

工数が合計 0.02 人時間発生した。製品ベンダーに 0.02 人時間発生しており、サプライヤー、ユーザーの工数は発生していない。

¹⁴⁴ 本実証における外部からの簡易的な確認であることに留意。

5.4.4. 効果比較評価

5.4.4.1. コスト低減効果

実証項目別のコスト低減効果は以下のとおり。

- 初期導入時の体制構築・環境整備コスト
 - [2] SBOM 有り（手動生成）の場合 148.49 人時間、[3] SBOM 有り（自動）の場合 126.42 人時間、[4] SBOM 有り（自動・有償ツール）の場合 7.67 人時間の増加が確認された。

表 5-16 コスト低減効果比較

シナリオ	工数	コスト低減効果
[1] SBOM 無し（独自形式の部品情報1階層のみ）	1.83 人時間（内訳 環境整備：1.33 体制構築・学習：0.5） ・サプライヤー：0 ・製品ベンダー：1.83（内訳 環境整備：1.33 体制構築・学習：0.5） ・ユーザー：0	—
[2] SBOM 有り（手動生成）	150.32 人時間（内訳 環境整備：102.09 体制構築・学習：48.23） ・サプライヤー：62.72（内訳 環境整備：41.92 体制構築・学習：20.80） ・製品ベンダー：62.72（内訳 環境整備：41.92 体制構築・学習：20.80） ・ユーザー：24.88※（内訳 環境整備：18.25 体制構築・学習：6.63） ※製品ベンダーから共有された SBOM を活用するための、脆弱性特定自動化ツールの導入工数 ※調査・試行錯誤による工数を含む。学習効果による短縮可能性がある （上記は SBOM 作成方法の探索・試行錯誤の時間を多く含む。）	-148.49 人時間（内訳 環境整備：-100.76 体制構築・学習：-47.73） ・サプライヤー：-62.72（内訳 環境整備：-41.92 体制構築・学習：-20.80） ・製品ベンダー：-60.89（内訳 環境整備：-40.59 体制構築・学習：-20.30） ・ユーザー：-24.88※（内訳 環境整備：-18.25 体制構築・学習：-6.63）

シナリオ	工数	コスト低減効果
[3] SBOM 有り (自動)	128.25 人時間 (内訳 環境整備 : 71.08 体制構築・学習 : 57.17) ・サプライヤー : 51.68 (内訳 環境整備 : 26.42 体制構築・学習 : 25.27) ・製品ベンダー : 51.68 (内訳 環境整備 : 26.42 体制構築・学習 : 25.27) ・ユーザー : 24.88 (内訳 環境整備 : 18.25 体制構築・学習 : 6.63) ※ライセンス特定ツールの工数 (14.67 人時間) については、OSS Review Toolkit のインストールと動作確認 (未完) の工数と、FOSSology の作業手順書作成の合算値を使用している。 (FOSSology にインストール作業が発生しなかったため) ※学習効果による短縮可能性がある (上記はインストール方法の探索・試行錯誤の時間を多く含む。)	-126.42 人時間 (内訳 環境整備 : -69.75 体制構築・学習 : -56.67) ・サプライヤー : -51.68 (内訳 環境整備 : -26.42 体制構築・学習 : -25.27) ・製品ベンダー : -49.85 (内訳 環境整備 : -25.09 体制構築・学習 : -24.77) ・ユーザー : -24.88 (内訳 環境整備 : -18.25 体制構築・学習 : -6.63)
[4] SBOM 有り (自動・有償ツール)	9.5 人時間 (内訳 環境整備 : 3.0 体制構築・学習 : 6.5) ・サプライヤー : 0 ・製品ベンダー : 0 ・ユーザー : 9.5 (内訳 環境整備 : 3.0 体制構築・学習 : 6.5)	-7.67 人時間 (内訳 環境整備 : -1.67 体制構築・学習 : -6.00) ・サプライヤー : 0 ・製品ベンダー : 1.83 (内訳 環境整備 : 1.33 体制構築・学習 : 0.5) ・ユーザー : -9.5 (内訳 環境整備 : -3.0 体制構築・学習 : -6.5)

● 運用時のコスト

➤ SBOM 作成コスト

◆ 作成工数の計測

[2] SBOM 有り (手動生成) の場合 0.46 人時間、[3] SBOM 有り (自動) の場合 0.42 人時間、[4] SBOM 有り (自動・有償ツール) の場合 0.76 人時間の削減効果が確認された。

表 5-17 SBOM 作成コストの比較

シナリオ	工数	コスト低減効果
[1] SBOM 無し (独自形式の部品情報 1 階層のみ)	0.79 人時間 (内訳 作成 : 0.35 精査 : 0.44) ・サプライヤー : 0.35 (内訳 作成 : 0.35 精査 : 0) ・製品ベンダー : 0.44 (内訳 作成 : 0 精査 : 0.44) ・ユーザー : 0	—
[2] SBOM 有り (手動生成)	0.33 人時間 (内訳 作成 : 0.33 精査 : 0) ・サプライヤー : 0.33 (内訳 作成 : 0.33 精査 : 0) ・製品ベンダー : 0 ・ユーザー : 0	0.46 人時間 (内訳 作成 : 0.02 精査 : 0.44) ・サプライヤー : 0.02 (内訳 作成 : 0.02 精査 : 0) ・製品ベンダー : 0.44 (内訳 作成 : 0 精査 : 0.44) ・ユーザー : 0
[3] SBOM 有り (自動)	0.37 人時間 (内訳 作成 : 0.03 精査 : 0.34) ・サプライヤー : 0.18 (内訳 作成 : 0.02 精査 : 0.17) ・製品ベンダー : 0.18 (内訳 作成 : 0.02 精査 : 0.17) ・ユーザー : 0	0.42 人時間 (内訳 作成 : 0.32 精査 : 0.10) ・サプライヤー : 0.17 (内訳 作成 : 0.33 精査 : -0.17) ・製品ベンダー : 0.26 (内訳 作成 : -0.02 精査 : 0.27) ・ユーザー : 0

シナリオ	工数	コスト低減効果
[4] SBOM 有り (自動・有償ツール)	0.03 人時間※ (内訳 作成：0.03 精査：9.83) ・サプライヤー：0 ・製品ベンダー：5.83 (内訳 作成：0 精査：5.83※) ・ユーザー：4.03 (内訳 作成：0.03 精査：4.00※) ※精査工数の内訳は下記のとおり。 このうち、新たに検出された OSS 部品の精査工数は他のシナリオ ([1][2][3]) で該当する作業なし。 ・製品ベンダー： 検知結果確認：0.00 新たに検出された OSS 部品の精査・ユーザー対応：5.83 ・ユーザー： 検知結果確認：0.17 新たに検出された OSS 部品の精査・製品ベンダー確認：3.83	0.76 人時間 (内訳 作成：0.32 精査：-9.39) ・サプライヤー：0.35 (内訳 作成：0.35 精査：0) ・製品ベンダー：-5.39 (内訳 作成：0 精査：-5.39※) ・ユーザー：-4.03 (内訳 作成：-0.03 精査：-4.00※)

➤ SBOM 活用のコスト

◆ 部品管理工数 (脆弱性管理、ライセンス管理) の計測

[2] SBOM 有り (手動生成) の場合 0.71 人時間、[3] SBOM 有り (自動) の場合 0.71 人時間、[4] SBOM 有り (自動・有償ツール) の場合 0.76 人時間の削減効果が確認された。

表 5-18 SBOM 活用コストの比較

シナリオ	工数	コスト低減効果
[1] SBOM 無し (独自形式の部品情報 1 階層のみ)	0.79 人時間 ・サプライヤー：1.87 ・製品ベンダー：0 ・ユーザー：0	—
[2] SBOM 有り (手動生成)	0.08 人時間 ・サプライヤー：0.03 ・製品ベンダー：0.03 ※ ・ユーザー：0.02 ※FOSSology の仕様 (SBOM 読込不能、ソースコードスキャンが必要) により、製品ベンダーがライセンス特定を実施必要なため	0.71 人時間 ・サプライヤー：1.84 ・製品ベンダー：-0.03 ※ ・ユーザー：-0.02
[3] SBOM 有り (自動)	0.08 人時間 ・サプライヤー：0.03 ・製品ベンダー：0.03 ※	0.71 人時間 ・サプライヤー：1.84 ・製品ベンダー：-0.03

シナリオ	工数	コスト低減効果
	・ユーザー：0.02 ※FOSSology の仕様（SBOM 読込不能、ソースコードスキャンが必要）により、製品ベンダーがライセンス特定を実施必要なため	※ ・ユーザー：-0.02
[4] SBOM 有り（自動・有償ツール）	0.03 人時間 ・サプライヤー：0 ・製品ベンダー：0 ・ユーザー：0.03	0.76 人時間 ・サプライヤー：1.87 ・製品ベンダー：0 ・ユーザー：-0.03

5.4.4.2. 脆弱性リスク低減効果

- 脆弱性対応期間の短縮効果の評価

脆弱性対応期間が最も大きいのは[1]（独自形式の部品情報）だった。SBOM ツール（有償、無償）を用いることにより、工数をかけずに頻繁に脆弱性検索が可能になるため、脆弱性の認識までの期間が短縮されるほか、ソフトの利用を中断するなどの暫定策で発見するまで防げなかった可能性のあるリスクを低減することが可能となる。一方、脆弱性修正の所要期間の短縮効果は確認されなかった。

表 5-19 リスク低減効果の比較

シナリオ	期間	リスク低減効果
[1] SBOM 無し（独自形式の部品情報 1 階層のみ）	3 か月と 3-5 日間 リスク特定：3 か月 修正までの期間：3-5 日間	—
[2] SBOM 有り（手動生成）	3-5 日間 リスク特定：0 日間 修正までの期間：3-5 日間	3 か月 リスク特定：3 か月 修正までの期間：0 日
[3] SBOM 有り（自動）	3-5 日間 リスク特定：0 日間 修正までの期間：3-5 日間	3 か月 リスク特定：3 か月 修正までの期間：0 日
[4] SBOM 有り（自動・有償ツール）	3-5 日間 リスク特定：0 日間 修正までの期間：3-5 日間	3 か月 リスク特定：3 か月 修正までの期間：0 日

- 脆弱性のリスク低減効果の評価

シナリオ[1]～[3]では、特定された部品は 7 件、シナリオ[4] SBOM 有償ツールでは、追加で部品 11 件が特定され、全体として特定された部品は 18 件であった。いずれのシナリオでも実証期間においては脆弱性が検出されなかったが、シナリオ[4]は脆弱性管理の対象となる部品の対象範囲が広いため、将来にわたって脆弱性が発見され

る確率はシナリオ[1]～[3]と比較して高まることから、将来にわたる脆弱性リスク¹⁴⁵（＝（残存する可能性）×（脆弱性の影響度））は低減すると言える。

表 5-20 部品管理対象件数・脆弱性発見数の比較

シナリオ	部品管理対象件数	実証期間の脆弱性発見数
[1] SBOM 無し（独自形式の部品情報 1 階層のみ）	7	0
[2] SBOM 有り（手動生成）	7	0
[3] SBOM 有り（自動）	7	0
[4] SBOM 有り（自動・有償ツール）	18	0

- 脆弱性特定コストの低減効果の評価
[2] SBOM 有り（手動生成）の場合 0.62 人時間、[3] SBOM 有り（自動）の場合 0.62 人時間、[4] SBOM 有り（自動・有償ツール）の場合 0.68 人時間の削減効果が確認された。

表 5-21 脆弱性特定コストの低減効果の評価

シナリオ	工数	コスト低減効果
[1] SBOM 無し（独自形式の部品情報 1 階層のみ）	0.70 人時間 ・ サプライヤー：0.7 ・ 製品ベンダー：0 ・ ユーザー：0	—
[2] SBOM 有り（手動生成）	0.08 人時間 ・ サプライヤー：0.02 ・ 製品ベンダー：0.02 ・ ユーザー：0.02	0.62 人時間 ・ サプライヤー：0.68 ・ 製品ベンダー：-0.02 ・ ユーザー：-0.02
[3] SBOM 有り（自動）	0.08 人時間 ・ サプライヤー：0.02 ・ 製品ベンダー：0.02 ・ ユーザー：0.02	0.62 人時間 ・ サプライヤー：0.68 ・ 製品ベンダー：-0.02 ・ ユーザー：-0.02
[4] SBOM 有り（自動・有償ツール）	0.02 人時間 ・ サプライヤー：0 ・ 製品ベンダー：0 ・ ユーザー：0.02	0.68 人時間 ・ サプライヤー：0.7 ・ 製品ベンダー：0 ・ ユーザー：-0.02

5.4.4.3. ライセンス管理効果

- ライセンス違反特定コストの低減効果の測定
[2] SBOM 有り（手動生成）の場合 1.14 人時間、[3] SBOM 有り（自動）の場合 1.14 人時間、[4] SBOM 有り（自動・有償ツール）の場合 1.15 人時間の削減効果が確認さ

¹⁴⁵ リスクは、（事故の発生確率）×（事故の影響度）として定義される。本実証において、将来に渡り脆弱性を特定できる可能性（確率）が高まることから、脆弱性リスクが低減すると言える。

れた。

表 5-22 ライセンス違反特定コストの低減効果の測定

シナリオ	工数	コスト低減効果
[1] SBOM 無し（独自形式の部品情報 1 階層のみ）	1.17 人時間 ・ サプライヤー：1.17 ・ 製品ベンダー：0 ・ ユーザー：0	—
[2] SBOM 有り（手動生成）	0.03 人時間 ・ サプライヤー：0.02 ・ 製品ベンダー：0.02 ・ ユーザー：0	1.14 人時間 ・ サプライヤー：1.15 ・ 製品ベンダー：-0.02 ・ ユーザー：0
[3] SBOM 有り（自動）	0.03 人時間 ・ サプライヤー：0.02 ・ 製品ベンダー：0.02 ・ ユーザー：0	1.14 人時間 ・ サプライヤー：1.15 ・ 製品ベンダー：-0.02 ・ ユーザー：0
[4] SBOM 有り（自動・有償ツール）	0.02 人時間 ・ サプライヤー：0 ・ 製品ベンダー：0.02 ・ ユーザー：0	1.15 人時間 ・ サプライヤー：1.17 ・ 製品ベンダー：-0.02 ・ ユーザー：0

- ライセンス違反のリスク低減効果の測定

本実証では、[2] SBOM 有り（手動生成）の場合、[3] SBOM 有り（自動）の場合における、SBOM を読み込んでライセンスを検出可能なツールが確認されなかった。[4] SBOM 有り（自動・有償ツール）においては、ライセンスの誤検出が 1 件確認された。したがって、SBOM によるライセンス違反のリスク低減効果の測定に至っていない。

5.4.5. 部品管理の比較評価の留意点・制約事項

本実証における SBOM による比較評価の留意点・制約事項として、以下の点が挙げられる。

1. サプライチェーン全体でソースコード・設定ファイルが共有されていること。

部品管理の対象とした GARDEN Scenario Analyzer 自体が OSS であったことから、製品ベンダー・サプライヤー・ユーザーに関わらずソースコード・設定ファイルが無償ツールに入力することで SBOM を作成できる状態にあった。

そのため、SBOM 共有の有無による SBOM 作成・活用コストの差や、SBOM 生成・活用主体別での SBOM 生成・活用コストの差が、産業への SBOM 導入時の実態より過小評価されている可能性がある。

2. 製品ベンダー・サプライヤーにおいて、精度の高い OSS 管理が実施されていること。

SBOM 作成に使用される設定ファイル（"requirements.txt"）での OSS 部品の誤記載・記載漏れが無いため、製品ベンダー・ユーザーに関わらず設定ファイル（"requirements.txt"）

を確認することで正確に SBOM を作成できる状態にあった。

そのため、SBOM 共有の有無による SBOM 作成・活用コストの差や、SBOM 生成・活用主体別での SBOM 生成・活用コストや脆弱性・ライセンスリスクの差が、産業への SBOM 導入時の実態より過小評価されている可能性がある。特に、OSS や脆弱性・ライセンス特定時の誤検知・検知漏れに伴うコスト・リスクの測定に影響を及ぼしていると想定される。本結果の背景として、GARDEN Scenario Platform の開発の大部分が特定サプライヤーに集中しており、精度の高い OSS 管理が比較的容易な環境だったことが挙げられる。GARDEN Scenario Platform の開発に関与したサプライヤーは 1 社のみであり、当該サプライヤーが大部分の開発を担っていたため、当該サプライヤーから情報収集をすることで大部分の使用 OSS が把握可能な状態であった。したがって、製品ベンダーとサプライヤー間、又は複数のサプライヤー間で開発範囲が分散している場合の SBOM 生成・活用コストや脆弱性・ライセンスリスクについても実証が必要と考えられる。

3. 実証において生成された SBOM が、大統領令の最小要件、及び SPDX 等の国際規格で定められたすべての情報項目を有していないこと。

本実証において想定された SBOM 活用方法は、ツールを用いた脆弱性・ライセンスの自動特定のみであった。その結果、OSS 脆弱性管理ツール Grype が、CPE（大統領令の最小要素における「その他の一意の識別子」に該当）のみ記載された SBOM であれば読込可能だったため、本実証の条件下では大統領令の最小要件及び SPDX 等の国際規格を完全に満たした SBOM が不要であった。（なお、SBOM からライセンス検出が可能なツールは、本実証では確認されなかった。）

そのため、本実証における SBOM 生成コストは、大統領令の最小要件、及び SPDX 等の国際規格への準拠を求める主体を含むサプライチェーンでの SBOM 生成コストを十分に反映できていない可能性がある。本実証で必要とされた SBOM の情報項目は 1 項目である一方、大統領令の最小要件に準拠する場合は 7 項目、SPDX 規格に完全に準拠する場合は 70 以上の項目を作成必要となるため、産業への SBOM 導入時における SBOM 生成コストは本実証結果より増大する可能性がある。

4. SBOM からライセンス検出が可能な無償ツールが確認されていること。

今回ライセンス検出ツールとして採用した FOSSology は、SBOM を入力できず、ソースコードのヘッダーにおけるライセンス関連の記述を読み取ることでライセンスを特定していることが実証結果より判明した。そのため、SBOM からのライセンス検出可否の実証に至っていない。

5. 環境整備において、ツール導入時に利用可能な情報が限定的なこと。

今回の実証で採用した SBOM 生成・活用ツールにおいては、インストール手順・使用方法を公開情報から把握することが困難な場合があった。その結果、限定された情報から試行錯誤でインストール手順・使用方法を把握する必要があり、SBOM 生成・活用ツール使用時の環境整備工数の増大につながった。そのため、環境整備工数が過大評価されている可能性がある。

6. 本実証の評価結果に関する限界

本実証は、対象ソフトウェアは1事例についての事例に依存する点に留意する必要がある。また、プロジェクトがパッケージマネージャーで少数のOSSを使っているシンプルな形態であるという特異性や脆弱性のリスクに関する定量的な評価が難しい点にも留意が必要である。

表 5-23 環境整備において発生した障害・対応内容・得られたノウハウ

障害	対応内容	ノウハウ
インストール時にエラーが発生 (Scancode-Toolkit)	情報収集可能な範囲でどのような動作環境か確認のうえ、様々なインストール手法を実施。 エラーメッセージから適切な手法が何かを推察しながら進めた。	動作環境を確認してから必要なソフトをインストールするのが効率的。
読み込むファイルが限定されておりスキャンできない (Vuls)	今回の実証には実施不可能なため、実証を断念。	読み込み可能なファイル・スキャン方式が実証対象の製品に適しているかを事前精査するのが望ましい。
ライセンス検知漏れ (Black Duck)	以前から管理しているライセンス一覧と突合し、異なるライセンスがないか確認。ツールベンダーに問い合わせて間違いを指摘のうえ、データベースの情報を修正いただいた。	ツールのスキャン結果をただ信じるのではなく、人手による検証も必要と理解してライセンス管理を行うのが望ましい。

5.5. 実証成果と課題の整理

本実証を通じて、実際のソフトウェア製品及びそのサプライチェーン（サプライヤー、ベンダー、ユーザー企業）において、SBOMの作成、共有、活用（脆弱性管理、ライセンス管理）の一連の業務を行うことで、SBOMに関わる主なコスト・効果の項目について特定・確認することができた。それに基づき、SBOMのメリット等と課題について分析・整理する。

5.5.1. SBOMのメリット等

5.3.1に示すとおり、SBOMの有効性及び費用対効果の観点で、SBOMのメリット等について整理する。

5.5.1.1. SBOMの費用・効果の評価とメリット

5.4.2及び5.3.3の結果から本実証で比較評価の対象とした4つのシナリオに関して、費用

と効果に関わる主な情報を要約したものが以下の表である。

表 5-24 SBOM 実証シナリオにおける主な情報の整理

シナリオ	初期工数		運用工数				ツール費用 (初期・運用)
	環境整備 工数	体制構築・学 習工数	SBOM(部品情報)作成工数		脆弱性特 定工数	ライセンス 特定工数	
			作成工数	精査工数			
[1] 従来の 部品管理 (独自形式)	1.3 人時間(フ ォーマットの 定義)	0.5 人時間 (フォーマッ ト・運用の連 携)	3.0 分/部品	3.8 分/部品(OSS 一 覧のレビュー)	6 分/部品 (頻度:1.5 か月)	10 分/部品 *別途法務チ ェックに 2-5 営業日	(なし)
[2] SBOM (手動作成)	102.1 人時間 (活用ツ ール導入・処理 調査、テンプ レート整備) * 全社の合計値	48.2 人時間 (SBOM 作成 方法の学習、 作業手順書作 成) *全社の合 計値	2.9 分/部品	(なし) *製品ベンダーが手 動作成した部品情 報が正確と仮定	0 分/部品 (頻度:即時)	0 分/部品 *別途法務 チェックに 2-5 営業日 *ライセン スの検出漏 れが発生	
[3] SBOM (無償ツ ール)	71.1 人時間 (作成・活用ツ ール導入) *全 社の合計値	57.2 人時間(作 業手 順 書 作 成、学習) *全 社の合計値	0.2 分/部品	1.4 分/部品(検知結 果確認)			
[4] SBOM (有償ツ ール)	3.0 人時間 (商談、スキャ ン用アプリ導 入)	6.5 人時間 (学習、開発 元問い合わせ)	0.3 分/部品	1.4 分/部品(検知結 果確認)	0 分/部品 (頻度:即 時)	0 分/部品 *別途法務 チェックに 2-5 営業日	ユーザー数 やデータ量 等に応じた ライセンス 費用が発生
				(81.5 分/部品(コード スキャンにより新 たに検知した部品 に関する精査及び ベンダー確認))			

この実証結果から、この比較条件では、SBOM は導入のための初期工数（環境整備、学習等）が大きい、運用工数（SBOM 作成、活用）はツールの活用により従来の手作業部品管理に比べ小さい結果となり、管理対象のソフトウェア部品が多いほど、SBOM 導入効果が大きくなることが確認できた。

脆弱性特定をツールで自動化することで、都度の確認にかかる工数を増やさず確認頻度を多く設定できるため、脆弱性情報の発表から特定までのリードタイムの短縮と工数の削減に繋がると考えられる。

本実証事例では、部品あたりの脆弱性特定、ライセンス管理の工数低減効果から、部品数が 355 件以上で、全体として無償の OSS ツールを用いた SBOM による部品管理が手作業による従来の部品管理と比較してコストが低減することが確認できた。

上表の脆弱性特定工数と頻度に基づき、SBOM ツールにより脆弱性発表から特定までのリードタイムを短縮可能であることが確認できた（手作業の場合は脆弱性特定業務の頻度に依存する）。

有償ツールでは、OSS の依存関係を解析し、構成ファイルから特定できなかった OSS による他の OSS の再利用も検知できた。それに伴い管理の対象となる部品の範囲が拡大し、かつ、検出された部品の精査など、無償ツールでは行っていない工数を計測していることか

ら、無料ツールとの工数比較には留意が必要である。

脆弱性が発見された場合、脆弱性の影響範囲の特定は、対策を検討するうえで重要であり、多くの工数を要するケースがある。本実証では、SBOM ツールと既存の構成管理ツール（トレサビリティ管理ツール ZIPC TERAS）を連携させることで、脆弱性の影響範囲特定の工数削減につながる可能性があることを検討することができた。

5.5.1.2. SBOM の有効性

SBOM の適用は、無条件に有効とは言えず、一定の条件の下で有効性が得られる。例えば、OSS から再帰的に利用される部品で改変されている部品（Snippet）まで特定する場合、自動ツールにおいても部品の検出漏れ、誤検知が生じるため、人手によるソースコードの解析が必要であり、自身が開発していない改変部品の有無を判断するためには、コードに対する深い分析と理解が必要であり、ソースコードを新規に開発する以上のコストが生じる可能性も否定できない。

本実証事例において、5.5.1.1 に示すコスト評価結果から、SBOM の有効性について、他のケースの導入判断にも参考となるように一般化して表現すると、以下のようにまとめることができる。

SBOM ツールを用いた部品管理は、以下の条件が成立する時、コスト面で有効と確認することができた¹⁴⁶。

条件 1：SBOM ツールの初期工数・費用が、脆弱性特定コストよりも相対的に低い¹⁴⁷。

条件 2：ツールで作成した SBOM の精度が、手作業による部品管理と同程度である¹⁴⁸。

条件 3：OSS 等のサードパーティ部品から再帰的に利用される部品管理は対象としない¹⁴⁹。

なお、条件 1 については、本実証では、部品管理数が 355 件以上あればコスト効率性が示されるが、初期コストが相対的に大きい場合、それ以上の部品管理数が無ければコスト効率性は得られない。条件 2 については、SBOM ツールの部品特定精度が低い場合には、精査のコストが上昇し、SBOM ツールの採算性は悪化する。条件 3 については、サードパーティ部品の再帰的部品を対象とする場合、精査コストが事例に依存して一概に比較評価できないため、ここでは対象外としている。

5.5.2. 検討すべき課題

本実証を通じて様々な課題を発見することができた。

(1) 無償ツールの成熟度の向上

¹⁴⁶ 条件を満たさない場合、SBOM 活用がコスト低減に繋がるとは必ずしも言えない点に留意が必要。

¹⁴⁷ 本実証では、部品数 535 件で SBOM 活用の方が有効であることが確認できたが、初期コストが大きい場合より多くの部品数でなければ採算が取れない。

¹⁴⁸ ツールにより検出した部品の精査にコストが過剰にかからないことを前提とする。

¹⁴⁹ サードパーティの再帰的な部品のツールによる検出のコストは事例に依存し大きくことなる。本実証では評価の対象外とした。

SBOM ツールを導入する際にはツールの環境整備や学習に工数が発生する。特に、無償ツールは成熟度の観点で以下のような課題がある：

- ・ ドキュメントやノウハウが不足しており導入の障害となる。
- ・ 機能・精度面で、再帰的な利用部品や改変コード (Snippet) の検知ができない
- ・ 読み込み可能な SBOM フォーマットに制限がある。
- ・ ライセンスの検知漏れが発生する等

このようなことから、ツールに関する使用方法、留意事項等のノウハウを整理した日本語ドキュメント整備のような対応が考えられる他、ツール自体の精度、機能の向上が求められる。

(2) ツールによる部品検出結果の精査

製品ベンダー以外が SBOM を作成する場合、ツールにより検出される OSS の再利用やソースコード改変部品などの精査に係る工数が大きくなる、もしくは精査が困難である。製品ベンダー自身が認識していない部品が検出された場合の管理責任が曖昧になりうる。

製品ベンダー（ベンダーやサプライヤー）自身がソフトウェア部品を特定し、標準的な SBOM フォーマットで共有することが、サプライチェーン全体における部品管理の効率化や責任の明確化につながると考えられる。

5.5.3. 今後の取組の方向性整理

得られた知見及び課題を踏まえて、今後取り組むべき課題の方向性について整理すると以下のようになる。

(1) SBOM 活用モデルの最適化

本実証では、特定のサプライチェーン構造のシナリオに限定して、SBOM の有効性について条件付きで示すことができた。SBOM の有効性は適用範囲により異なるため、SBOM の適用範囲の選択肢を整理し、どのような場合に、SBOM が有効であるか体系的に整理することで、効果的な SBOM 活用モデルを整理することが重要である。そのためには、SBOM の作成主体、作成対象範囲、作成手段等の選択肢を体系的に整理し、それらの組合せによりコスト効果を評価することで、効果的な SBOM 適用範囲を検討することが考えられる。産業分野によって規制や取引形態の相違があるため、産業分野の状況に応じた SBOM の費用・効果を計測評価することで効果的な活用モデルを整理することが求められる。

(2) SBOM 共有のための環境整備

多数のステークホルダーが効率的に SBOM を共有・活用する標準、取引契約等の環境やモデルを整備することが重要である。標準化については、各分野等において必要となる SBOM の項目、フォーマット、部品命名規則等の共通化することで、企業間の SBOM データのやり取りを効率化することが必要である。また、取引企業間の部品管理における責任の明確化、SBOM 作成・納品の要件化、費用負担に関する契約に基づく取り決めが重要である。そのためには取引契約書における契約条項の参考モデルを例示することで、取引契約を利用して SBOM を普及促進する環境整備が必要である。

(3) ツールの活用促進による効率化

SBOM ツールの導入方法や利用方法に係る情報やノウハウの整理と発信を通じて企業における導入促進と導入コストの低減を推進することが求められる。また、無償ツールなどにおいては機能及び精度について未成熟な点があるため、それらの向上が期待される。

(4) 国際的な基準との整合化

グローバルサプライチェーンにおいて国内外の基準や制度に相違があると、企業の負担となるため、基準、制度の国際整合化が必要である。そのためには、日米の政府による連携・協力関係を維持しつつ、日本の実証において得られた知見から日本側が妥当と考える基準について米国に提案していくことが重要である。

6. ソフトウェアタスクフォースの運営

本項目では、上記 2,3,4,5 における調査結果・実証方針・実証結果等について議論する「サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース」（ソフトウェアタスクフォース）の運営とタスクフォースに向けた資料の作成支援を行った。

6.1. タスクフォースの実施概要

有識者（学識経験者やサイバーセキュリティ・ソフトウェア関連団体等を含む）やソフトウェア事業者等の委員によって構成されたソフトウェアタスクフォースは、産業サイバーセキュリティ研究会 WG1 の基に設置され、経済産業省によって開催されており、本事業ではその運営を行った。ソフトウェアタスクフォースでは、我が国のソフトウェア分野におけるサイバーセキュリティ向上策について検討が行われ、2021 年度にて検討した主な議題は以下の 2 点である。

- ・ 国内での SBOM 活用促進に向けた実証
- ・ OSS 管理手法に関する事例集の拡充

6.1.1. 第 5 回ソフトウェアタスクフォース

第 5 回ソフトウェアタスクフォースでは、最近のインシデント事例・ソフトウェア管理等に関する諸外国の取組状況・国内での SBOM 活用促進に向けた実証・OSS 管理手法に関する事例集の拡充の 4 点について事務局より報告された。自由討議では、国内での SBOM 活用促進に向けた実証に関して、SBOM 実証における方針や確認事項等について議論された。また、OSS 管理手法に関する事例集の拡充に関して、事例集から得られた課題等について議論された。

今後の SBOM に関する検討課題として、制度普及に向けた省庁連携や SBOM の政府利用などが挙げられた。また、SBOM の実証において、既存の SBOM 利用者の意見の取り込みや国際的な動向との連携を実施する必要性が挙げられた。

6.1.2. 第 6 回ソフトウェアタスクフォース

第 6 回ソフトウェアタスクフォースでは、近年のインシデント事例やソフトウェア管理等に関する諸外国の取組状況・国内での SBOM 活用促進に向けた実証・OSS 管理手法に関する事例集の拡充の 4 点について事務局より報告された。自由討議では、国内での SBOM 活用促進に向けた実証に関して、実証結果より得られた課題や課題に対応する今後の取組について議論された。

今後の SBOM に関する検討課題として、SBOM 制度の普及や国際的に連携されたソフトウェアのインフラ整備を実施していくうえで、大きな目標設定と省庁連携の必要性が挙げられた。その他に、SBOM が与えるメリットとソフトウェア説明における透明性の重要性を認知させるための仕組みづくりの重要性が挙げられた。

6.2. 実施スケジュール

2021 年度においては、以下のスケジュールでタスクフォースが実施された。また、各タスクフォースでは、以下の議事次第と資料にて議論された。

- ・ 第5回ソフトウェアタスクフォース
 - 日時：2021 年 10 月 29 日（金） 10:30~12:30
 - 開催方式：Web 開催
 - 議事次第
 1. 開会
 2. 事務局資料説明
 3. 自由討議
 4. 閉会
 - 配布資料
 1. 資料1 議事次第・配布資料一覧
 2. 資料2 委員名簿
 3. 資料3 サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォースの検討の方向性
- ・ 第6回ソフトウェアタスクフォース
 - 日時：2022 年 3 月 3 日（木） 11:00~12:45
 - 開催方式：Web 開催
 - 議事次第
 1. 開会
 2. 事務局資料説明
 3. 自由討議
 4. 閉会
 - 配布資料
 1. 資料1 議事次第・配布資料一覧
 2. 資料2 委員名簿
 3. 資料3 サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォースの検討の方向性

6.3. タスクフォースの実施結果の要点整理

各タスクフォースで有識者から得られた意見の概要について以下でまとめた。

6.3.1. 第5回ソフトウェアタスクフォースの意見概要

SBOM 活用の効果・コストに関する意見を多くいただいた。特にスケーラビリティを考慮した評価や検討の必要性について述べられた。また、SBOM 活用における責任分担として、OSS の SBOM に対する責任の所在を明示的に示していただきたいという意見があった。その他に、実証において、SBOM のツール・フォーマット・管理項目・粒度について確認いただきたいという意見があった。また、実証から得たノウハウの共有・可視化の必要性が述べられた。

これらの意見を踏まえたうえで、実証を実施し、5に示すとおりツール・フォーマット・管理項目・粒度について一定の成果を示すことができた。しかし、ノウハウの共有・可視化や各分野における検討が不十分であり、来年度以降の実証に向けての課題として挙げた。

6.3.2. 第6回ソフトウェアタスクフォースの意見概要

来年度以降の取組の方向性に関する意見を多くいただいた。特に政策を進めるための根拠として具体的な目標を「旗」として立てていただきたいという意見が多く委員から述べられた。また、来年度の実証に向けて、中小企業への普及・SBOM ツールの自動化等について意見が挙げられた。具体的には、中小企業の実証参加・脆弱性 DB の連携等が述べられた。普及に向けた取組として、人材育成・国際動向との連携等が挙げられた。特に、ソフトウェアセキュリティに関する意識向上を目的とした教育や海外における OSS 管理の取組との連携に関する意見を多くいただいた。

7. 今後の課題と取組案のまとめ

本報告書では、OSS の利活用に起因するセキュリティリスクを洗い出すとともに、国内外におけるソフトウェアの脆弱性の管理の取組動向や、OSS を含むソフトウェアを企業等で自社製品・サービスとして活用する際のセキュリティリスクに対応するための取組について調査を行い、多角的な観点から OSS を含むソフトウェアを安全に利活用できる基盤の構築に向けた検討や我が国のサイバーセキュリティ政策立案に資する調査を行った。また、SBOM の導入・活用に関する調査及び実証事業を通じ、費用対効果に関する評価分析、我が国における SBOM の普及への課題や効果等について整理を行った。

これらの調査結果に基づき、今後の課題と取組（案）について整理する。まず、本調査から抽出される主な課題や論点を以下にまとめる。

SBOM 実証及び各企業へのヒアリング結果

- SBOM 導入のための初期工数は発生するが、ツール活用により運用工数が低減。管理対象のソフトウェアが多いほど効果が大きくなりうる。
- ツールによる脆弱性特定のリードタイムの短縮効果を確認。特に無償ツールは導入工数が大きく機能・精度も不十分。ノウハウ共有やツール自体の精度、機能向上が必要である。
- 開発者自身が標準的な SBOM フォーマットで部品情報を共有することが有効だが、部品粒度等を揃える（要件化する）必要がある。
- OSS の再利用等も含め管理する場合のコスト（精査工数等）が大きい。
- 管理対象のソフトウェアやステークホルダーが多い場合の実証や、繰り返し発生する脆弱性対応工数の評価は未実施。

国内外の動向等（調査結果）

- 大統領令に基づき、米国政府調達における SBOM 提供の義務化に向けた動きが進展。SBOM の最小要素が公表された。
- CISA のイベント SBOM-a-rama では、今後の主要な課題が抽出された。（SBOM 生成に必要なデータの提供方法、SBOM 共有・活用ツール、各国の規制基準への適用、SBOM 共有にあたっての信頼性や完全性の確保 等）
- 医療機器分野では、IMDRF ガイダンスに基づき、国内でも規制改正による SBOM 提供義務化の動きがある。

第5回までのタスクフォースにおける議論（抜粋）

- 効果・コスト評価（複数社の連携、システム規模の差、SBOM の更新）
- SBOM 共有時のコストや責任の分担、契約の整理
- 自動化・ツール（費用、精度、信頼性、相互運用性）
- メリットを享受できる活用モデルの検討
- 脆弱性の対応要否判断
- SBOM フォーマットや必要項目、管理単位、粒度の整理、標準化
- 中小企業への配慮、巻き込み
- ノウハウの共有（プラクティス集、ガイドライン）

- 政府機関での活用（実証の実施、政府統一基準）

第6回タスクフォースにおける意見・議論

- SBOMの受益者、作成者の能力、制度との関係などの整理が必要
- 免責のガイドラインがあるとよい
- VEXなど脅威情報に基づく対策管理が必要
- デジタル庁、政府調達基準との関係性の検討
- 著作権法、企業秘密保持などの関係性や対処法の検討
- OSSディストリビューターとの連携の必要性
- ツール比較選択ガイダンス
- ツールによる自動化について国際的な整合性の確保
- 米国のSBOM最小要素の要件、粒度の整合化の必要
- ソフトウェア技術者のコンピテンシーの向上

以上の調査結果及び意見等に基づき、今後の課題を整理すると以下のようにまとめることができる。

表 7-1 今後の課題

課題	内容
<u>1. SBOM 活用モデルの最適化</u>	SBOMの作成主体や作成対象範囲、作成手段等によりコストや効果が異なり、産業分野によっては規制等の動きもあるため、産業分野の状況に応じたSBOMの効果的な活用モデルを整理する必要がある。
<u>2. SBOM 共有のための環境整備</u>	● 多数のステークホルダーがSBOMを共有・活用するケースの実証が必要である。 ● 共有時の効果を最大化するため、SBOMの項目、粒度、フォーマット、部品命名規則などの標準化が必要である。 ● 契約、責任、費用負担の整理が必要である。
<u>3. ツールの活用促進による効率化</u>	● SBOMツールの導入や利用方法に係る情報発信、ノウハウの共有による導入工数の低減が必要である。 ● ツール自体の機能及び精度向上が必要である。
<u>4. 国際的な基準との整合性確保</u>	グローバルサプライチェーンにおいて国内と海外の整合性を確保しつつ効率的に部品管理を行うためには、国内外の基準の整合化が必要である。
<u>5. 人財育成・意識啓発</u>	SBOMに関するソフトウェア技術者のコンピテンシーや意識の向上が必要である。

また、これらの課題を解決する取組策と対応関係を整理すると以下ようになる。

表 7-2 今後の課題と取組策の対応関係

令和4年度以降の取組	対応する課題	実施時期	実施主体
① 実証によるコスト・効果の評価と論点整理の継続 規制等が進む分野や SBOM 導入効果が大きい分野等を候補に、複数分野で実証を行うとともに、コスト・効果の評価、個別の課題等に係る知見を共有。成果を他の取組においても活用。	1,2	R4 年度	国・民間
② SBOM の効果的な活用モデルの検討 各産業分野における規制の内容やリスク等を踏まえた効果的な活用モデル及びステークホルダーの合意形成の在り方、政府機関での活用などについて検討。	1,2	R4 年度以降	国
③ SBOM 共有に関する取引モデルの検討 サプライチェーンの部品管理に関する責任、部品情報共有の要件、費用負担等、法制度との関係性、取引契約における論点を整理。産業分野ごとの取引契約書モデル、免責を含むガイドライン、OSS コミュニティ、ディストリビューターとの連携方策の検討。	2	R4 年度以降	国・民間
④ SBOM に関するノウハウ共有 実証で得た SBOM ツールのノウハウ、コスト情報等についてガイド／手引きを作成。 NTIA のプレイブック等との対応も意識し、調整が必要な事項を整理。	3	R4 年度	国
⑤ SBOM 自動化・共有に向けた技術的な検討 SBOM ツールの選択肢等の調査、SBOM に係る基準や国際標準等への対応、構成管理ツールとの統合、脆弱性 DB におけるソフトウェアの識別子の整備、SBOM 信頼性確保等に向けた検討や研究開発。	1,3,4	R4 年度	国・民間
⑥ 国外との制度調和 米国と意見交換や成果の共有ができる関係を維持。制度等、必要な調整を実施。	4	R4 年度以降	国
⑦ 人財育成・意識啓発 ②,③,④などの成果を活用しつつ、独立行政法人や社会人大学院講座(文科省施策との連携)などによるソフトウェア技術者のコンピテンシーの向上と意識啓発を図る。	5	R4 年度以降	国・民間

8. 付録

8.1. タスクフォース議事要旨

8.1.1. 第5回ソフトウェアタスクフォース

1. 日時・場所

日時：2021年10月29日（金）10:30～12:20

場所：オンライン開催

2. 出席者

委員：土居委員（座長）、出雲委員、伊藤委員、猪俣委員、大場委員（代理：遠藤様）、木谷委員、下村委員、関委員、高田委員、高橋委員、寺田委員、野山委員、萩原委員、平田委員、松岡委員、渡辺委員

オブザーバ：内閣官房 内閣サイバーセキュリティセンター、厚生労働省、一般社団法人 日本医療機器産業連合会

経済産業省：大臣官房 江口サイバーセキュリティ・情報化審議官、商務情報政策局 奥田サイバーセキュリティ課長、佐藤サイバーセキュリティ課企画官、入江サイバーセキュリティ課長補佐

3. 配付資料

資料1 議事次第・配布資料一覧

資料2 委員名簿

資料3 サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォースの検討の方向性

4. 議事内容

事務局から資料3に基づき説明した後、自由討議を行った。委員等からの意見は以下のとおり。

・ SBOM の効果・コストについて

- SBOM 活用のメリットについて、一社に対して効果がある部分、複数社が連携して取り組むことにより効果が上がる部分、官民連携で効果が上がる部分を分析できると良い。
- コンポーネントの把握、対応スピード、効率性の3つの観点での向上が見込まれ、OSSを含めて SBOM を活用した構成管理が一般的となる可能性。実証で国内の開発に関し構成管理を含めたものへのレディネスをどう作るか、というところが見えてくると良い。
- コストに関する課題については、規模によるスケーラビリティの方向性が示せると良い。命名規則については何らかの形で統一されることが望まれる。OSSのコミュニティがなくなっているケースなど、長期間修正されないマイナーな脆弱性が存在し、アプリケーションの脆弱性対応度が100%とならないケースも存在する。
- どれほどのシステム規模であれば SBOM の効果があるかを示せると良い。大規模システムであれば直感的に効果があることが分かるが、SBOM が効果を発揮するシステム規模の境目があるだろう。

- コストや責任の分担について
 - サプライチェーン全体としてメリットがあることを示し、コストをどのように分配するかを考えると良い。
 - 脆弱性管理のメリットを分析するにあたって、保守契約がどうなっているのか、曖昧であればどのような解決方法が考えられるかを検討するのが良いのではないか。また、個社が SBOM に関する情報を出せないという問題もある。契約の問題を打破しないと SBOM 活用には結びつかないので、実証で検討いただけると良い。
 - 将来的には、OSS にも SBOM が付属され、製品メーカーはその OSS のコンポーネント解析を実施せず、自身の SBOM と組み合わせて提供できる形が理想。ただし、コミュニティに責任を問わない形で、いかに OSS の SBOM を信頼するかは議論の対象。
 - 最終製品ベンダーが解析を実施する場合、OSS コミュニティが SBOM を提出しても活用されない可能性。ソフトウェアと SBOM の品質保証を一体と考え、ソフトウェアベンダーの責任が問われない場合はその SBOM の責任も問われないなどの仕組みが望まれる。SBOM 生成に係る受発注者間の責任分担のパターンを検討できると良い。
- SBOM に関するノウハウの共有について
 - コストや効果の計測だけでなく、SBOM 利用の際の手間やノウハウも記録し、報告書で記載されると良い。また、SBOM に含まれるソフトウェアコンポーネントの名称や情報が一致しない場合、手動と自動を組み合わせたハイブリッドのケースが多いと考えている。そのようなケースにおける効果やノウハウを可視化できると良い。
 - 今回の実証と並行して、開発者が実際に試せるプラクティス集をオープンソースで公開、共有する等も大切な視点であるし、望まれる進め方の一つ。こうした開発者自らが手を動かせる点についても検討を進めることは、できることから始めるという観点でも、日本での開発者の態勢の構築していく観点でも重要な課題。
 - 以前同様の実証を行った際、既存の資産管理ツールを使っていることが多く、SBOM ツールを活用する際にシステムやデータの移行をどのように実施するかが難しかったため、何らかの指針があると良い。
- SBOM のツールについて
 - SBOM 活用はツールに依存する部分が大きく、ツール自体の信用性や確からしさを示せると良い。SBOM の活用を促進するという観点では、ツールも OSS であると普及が促進するのではないか。
 - SBOM 実証に関して、ソフトウェア管理に関する取組は進みつつある。実証において、自動化前提で進めないとサプライチェーンの末端の企業は対応が困難。自動化ツールを予算の範囲で試していただきたい。
 - IPA の JVN を改修し、SWID タグ、SPDX、CycloneDX など、ソフトウェアの識別子を登録するデータベースの整備について検討が行われている。脆弱性情報と脅威情報を一緒に送れるよう、フォーマット変換も検討中。ソフトウェアの識別子を使い、ツールベンダーに資産管理だけでなく脆弱性管理をしてほ

しい。ソフトウェアの識別子の管理課題などで連携できると良い。

- ツール間の互換性がなく、相互運用する際に、異なるツールを用いることが難しい。
- ・ 産業界における既存の取組について
 - 国内企業でも既に SBOM を導入している企業にヒアリングしながら、包括的な課題抽出をした方が良い。
 - どの程度の企業が構成管理を実施しているのか、各構成管理ツールの販売実績を確認することも一案。
- ・ 政府機関での SBOM 活用について
 - 実証に関して、政府機関で活用することも一案。その後、政府統一基準に SBOM の要件を入れ込むことを検討できると良い。現状は曖昧な部分が多いため、実証を通じて具体化することがまず必要。
- ・ その他
 - 現状でもソフトウェアの構成管理を独自に実施しているが、SBOM を利用した場合にどのようになるかは非常に興味深い。OSS を活用することでアップデートが頻繁になるほか、構成が複雑になる中で、いかに工数を削減するかが課題。グローバルの規定が厳しくなる中で、事業者の体力を奪うことがない施策になると良い。
 - 防衛産業では、業務発注に関わるすべてを監査するという形で進めているが、民間と齟齬をきたさないように進める必要があり、経済産業省においても、検討の範疇に入れる必要があると考えている。
 - 米国では、自動化が進まないと SBOM 導入が進まないというのが産業側のコンセンサス。既存のコミュニティベースでコンセンサスを醸成しつつ標準を開発することに加え、ユーザーによって SBOM の使い方は変わってくるため、様々なパターンを考えていかなければならない。日米間の情報共有ができると良い。
 - 日米協力について、NTIA のほか、CISA や NIST など複数の組織との関係構築を目指すのが良い。
 - 脆弱性情報は日々変わっており、ベンダー自身が情報を探して活用することは難しく、OSS コミュニティとのコネクションが重要。いかに連携していくかが課題か。

8.1.2. 第6回ソフトウェアタスクフォース

1. 日時・場所

日時：2022年3月3日（木）11:00～12:45

場所：オンライン開催

2. 出席者

委員：土居委員（座長）、出雲委員、伊藤委員、稲垣委員、猪俣委員、大場委員、木谷委員、下村委員、関委員、高田委員、高橋委員、寺田委員、野山委員、萩原委員、平田委員、松岡委員、渡辺委員

オブザーバ：内閣官房 内閣サイバーセキュリティセンター、警察庁、厚生労働省、

一般社団法人 日本医療機器産業連合会

経済産業省：大臣官房 江口サイバーセキュリティ・情報化審議官、商務情報政策局
奥田サイバーセキュリティ課長、佐藤サイバーセキュリティ課企画官、塚本サイバー
セキュリティ課補佐、石垣サイバーセキュリティ課補佐

3. 配付資料

資料1 議事次第・配布資料一覧

資料2 委員名簿

資料3 サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等
検討タスクフォースの検討の方向性

4. 議事内容

事務局から資料3に基づき説明した後、自由討議を行った。委員等からの意見は以下のとおり。

- ・ 来年度以降の取組の方向性について

- SBOMが社会で機能を発揮するには、時間軸、産官学、個々のユーザー、ベンダー、事業の受益者がなすべきことの整理、SBOM実装を支える人材の育成強化、社会制度との連携等、様々な課題を克服する必要がある、今の取組はその出発点である。SBOMによってリスク分析の基礎情報が共有され、リスクコミュニケーションが可能になり、サプライチェーンのリスクを統制できるようになることを踏まえれば、我々は、経済安全保障を含む我が国のセキュリティ政策全般を支える重要不可欠なインフラ構築を論じていることを自覚すべきと考える。よって、今後の我が国のセキュリティ政策全般を進めるための根拠として、3～5年でSBOMの社会実装に必要な具体的な目標を「旗」として立て、それに見合う予算を獲得し、誰の、何のために、どのような人材で、どのような社会制度と関連付けていくのか目的を明確にし、現実的に推進していただきたい。また、対象範囲とその根拠等も現実の分析を踏まえた意思決定が必要。浸透方法も様々な手法が考えられるため、どこを目指すか明示いただけるとよい。国際競争上も重要であり、日本が優位を取れるよう進められるとよい。
- 業界ごとにローカライゼーションするのではなく、全体として「旗」を立てる方向が望ましい。
- SBOMの運用に関し、ライセンスを確認するのは主に製品リリース前・開発中と考えられ、委託先とSBOMをやり取りしながらメーカーが最終確認するような運用イメージである。製品リリース後、脆弱性有無を正確に確認するためには、SBOMが最新であること（プログラム更新と同期）と、SBOMと突合する脆弱性情報（VEXなど）が最新であることが条件と考えられ、脆弱性をリアルタイムで確認するために必要な情報の管理方法や体制についても議論する必要。
- SBOMの必要性について、脆弱性やライセンスコンプライアンス等、負の側面の担保のほか、開発生産性の向上や脆弱性に関する問合せ対応の短縮化のような正のメリットも共有して普及を進められるとよい。
- 先行する米国等における権利関係の工夫、著作権法や企業秘密などをクリアする方法についても調査いただきたい。

- これまで各自が取り扱ってきた課題をソフトウェアの透明化で解決できることが、SBOM の基本概念であることを示していけるとよい。
- ・ 中小企業等への普及について
 - 中小企業への浸透についても今後検討していく必要があり、実証にも協力したい。来年度以降の対象の選定にあたって、中小のソフトウェアベンダーにも参加いただき、事例を作成できると良いのではないかな。
 - 中小企業の状況も考慮したうえで、SBOM の活用モデルの構築やノウハウ共有を行っていただきたい。特に SBOM 作成ツールに関するノウハウが重要である。
 - 今後 SBOM は必須になると考えているので、必要性をより強く示していただきたい。Log4j の脆弱性等では依存関係の情報を含む SBOM が必要になる一方で、中小企業では SBOM を手動で構築する労力を確保できないため、自動で確認できるツール等が必要。ツール利用に関する施策を、ナレッジを共有しスピーディに進められるとよい。「旗」を振る際も、中小企業を含めて実施いただけるとよい。
 - 中小のソフトウェア事業者における SBOM の認識状況やセキュリティ意識なども調査していただきたい。OSS を利用する危険性の認識度合いを確認できると次の取組も考えられる。
- ・ SBOM に必要な要素等について
 - どのような粒度で SBOM を作るべきかが重要。
 - 標準的なフォーマットの検討も重要。活用モデルと合わせて検討されると理解。
 - SBOM が改ざんされていないことを示す電子署名が必要ではないか。
- ・ SBOM ツールや自動化について
 - 資料で紹介された Log4j のような脆弱性も検出される必要。有償ツールを利用して厳格に管理している場合は、再帰的に Log4j を使っているケースも含めて迅速に脆弱性を特定でき、SBOM の効果を感じた。
 - 脆弱性のレポジトリとの連携も重要。脆弱性対策情報データベース JVN iPedia と SBOM における製品との対応付けについては従来から取り組んでおり、現状、DB に登録されているソフトウェアには製品 ID が振られているが、DB に登録されていない場合は製品 ID が無いことが課題。経済産業省とも別途相談して、検討状況を報告したい。
 - ツールによって検出精度に差があるため、どのようなツールでどこまで対応していれば自分に非が無いか、免責について具体的に記載されているガイドラインがあるとよい。
 - Cyclone DX のプロジェクトにおいて、脆弱性の発動条件を示す VEX というフォーマットが提唱されており、脆弱性の発生を点ではなく範囲で確認可能。VEX の活用による脆弱性確認の自動化の方向性も検討する必要。
 - 元々ライセンス管理をしていた IT 資産管理ツールベンダーが、脆弱性管理やセキュリティへも対応し始めているため協力できるとよい。
 - ライセンス購入以外にマネージドサービスとしてスポット利用可能なツールも作成されてきている。

- 米国ではツールによる自動化の議論が進んでいるため、国際的な平仄を合わせておくといよい。
- ・ OSS の管理等について
 - OSS のディストリビューターとの連携も重要。
 - 2022 年 1 月にホワイトハウスが OSS コミュニティや関連企業を集めて OSS を守るための会議を実施している。日本でも同様の取組を実施できるとよい。
 - OSS を有償で構築している人も多く、米国では OSS に対するファンドについても議論されている。また、Linux Foundation が実施したアンケートでは、8 割以上が米国大統領令に向けて体制を構築していると回答。Log4j の影響もあり、各企業はフォーマットの統一化や SBOM の普及を米国政府に求めている。
 - Linux Foundation が 2022 年 2 月に SBOM は必須となったとのアナウンスを出している。
 - OSS を作る側だけでなく、利用者側の義務も重要。ソフトウェア開発においても、プロジェクトマネジメント義務に関しユーザーの義務が理解される傾向にある。ユーザー側が実施すべき項目もあることを考慮いただきたい。
- ・ 人材育成について
 - 無償の SBOM ツールである Syft を利用した際、バージョン情報の結果が様々であり、最終的に手動で判断する必要があった。官民だけでなく、アカデミーを利用し人材育成を含むエコシステムを構築できるとよい。
 - SBOM は、中長期的にはソフトウェア技術者のコンピテンシーの問題。この問題を解決しない限り後追い対策になってしまう。ソフトウェア技術者に認識いただくためにも、IPA の情報処理技術者試験等に組み込めるとよい。
 - SBOM に関わる者の意識は、時間をかけて形成する必要がある、研究機関や高度教育機関の役割が重要。大学等におけるセキュリティの基礎教育や研究の体制に SBOM を組み込むよう文部科学省へも働きかけていただきたい。
- ・ 政府機関等における取組について
 - サイバーセキュリティ政策に関する日米の政府間対話でも、本内容について意見交換できるとよい。
 - 情報セキュリティ早期警戒パートナーシップで届けられたソフトウェアの脆弱性を JPCERT/CC が確認し、各ソフトウェアベンダーへ連絡する取組においても SBOM が活用できるのではないかな。
 - デジタル庁が実証候補に含まれていないが、影響力の大きいチャレンジは国で実施すべきではないかな。

令和 3 年度 サイバー・フィジカル・セキュリティ対策促進事業
SBOM を導入・活用するサプライチェーンモデルの構築に向けた
調査・実証事業

2022 年 3 月
株式会社 三菱総合研究所
デジタル・イノベーション本部
Tel: 03-6858-3578

OSS の利活用及びそのセキュリティ確保に向けた 管理手法に関する事例集

経済産業省 商務情報政策局
サイバーセキュリティ課

令和 3 年 4 月 21 日
令和 4 年 X 月 X 日 追加

目次

1. 目的	3
2. OSS の概要	4
2.1. OSS とは	4
2.2. OSS の利活用領域	6
2.3. OSS 利活用のメリット	8
2.4. OSS 利活用の留意点	9
2.5. OSS に関わる係争及びインシデント事例	10
2.6. OSS 利活用等に寄与する主な取組	12
3. 事例の整理方法	16
3.1. OSS を取り巻く商流・ステークホルダの整理	16
3.2. OSS の利活用に係る課題観点の抽出	19
3.3. 紹介事例一覧	22
4. 事例（ヒアリング調査）	25
4.1. トヨタ自動車 ～サプライチェーンにおけるソフトウェア使用状況把握～	25
4.2. ソニー ～各事業部による主体性のある取組～	33
4.3. オリンパス ～ヒヤリ・ハット事象を契機とした全社的取組～	37
4.4. 日立製作所 ～製品化の過程における徹底した OSS 管理～	44
4.5. オムロン ～PSIRT の連携を通じた OSS 対応～	48
4.6. 東芝 ～グループにおける一貫した OSS 対応体制～	51
4.7. デンソー ～サプライチェーン全体における最適な OSS 管理～	58
4.8. 富士通 ～部門横断の OSS 対応体制と全社統一的なソフトウェア管理～	63
4.9. 日本電気（NEC） ～事業部ごとの取組から全社的取組へ～	71
4.10. 日本電信電話（NTT） ～OSS サポートに係る適切な役割分担～	79
4.11. 匿名企業 A 社 ～OSS 選定基準の明確化とコミュニティ活動～	82
4.12. 匿名企業 B 社 ～グループ内 SIer を中心としたセキュリティ強化～	84
4.13. 損害保険ジャパン ～ソフトウェア部品構成表を活用した脆弱性管理～	86
4.14. Visional グループ ～自社状況に対して最適なツールの利用～	90
4.15. サイボウズ ～OSS エコシステムに貢献する OSS ポリシー～	94
4.16. 三菱電機インフォメーションシステムズ（MDIS） ～OSS の特性を踏まえたソリューション展開～ ...	98
4.17. OSSTech ～OSS を活用したソフトウェア管理～	101
4.18. ヤフー ～社員が正しく OSS を使うための体制整備～	103
4.19. ラキール ～OSS の選定会議と検証ツールの併用による開発と管理の効率化～	107
4.20. SCSK ～オープンソースソフトウェアへの取組～	110

5. 事例（文献調査）	121
5.1. マイクロソフト ～OSSに係るセキュリティリスク緩和策～	121
5.2. ザランド ～OSS プロジェクトの全社的な推進～	124
5.3. Linux Foundation とハーバード大学による Census II プロジェクトの予備的レポート ～アプリケーションに最も利用されている FOSS コンポーネントに関する調査～	127
6. まとめ	131

1. 目的

産業活動のサービス化に伴い、産業に占めるソフトウェアの重要性は高まっている。特に、近年は、産業機械や自動車等の制御にもソフトウェアの導入が進んでおり、汎用的なハードウェア上にシステムを構築し、ソフトウェアにより多様な機能を実現することで、様々な付加価値を創出していくことが期待されている。

なかでも、ソースコードが一般に公開され、商用か非商用かを問わずソースコードの利用・修正・再配布が可能なオープンソースソフトウェア（OSS）については、汎用ライブラリや Linux システム等を中心に、近年、企業の商用製品・サービスにも積極的に採用されており、今や OSS を用いずに製品・サービスを構築することは困難と言える。

ソフトウェアのセキュリティ確保に向けた取組として、産業サイバーセキュリティ研究会ワーキンググループ 1（制度・技術・標準化）が 2019 年 4 月に取りまとめた「サイバー・フィジカル・セキュリティ対策フレームワーク」（以下、「CPSF」という。）では、ソフトウェアの構成管理や、完全性確認等について言及しているものの、ソフトウェアの複雑化や OSS の利用拡大等に伴い、ソフトウェアのセキュリティをどのように維持し続けるのか、それをどのように確認するか具体的な方法までは明確化していない。海外では、米国商務省の電気通信情報局（NTIA）において、2018 年 7 月から Software Component Transparency という官民合同の検討体制を構築し、Software Bill of Materials（SBOM）を用いたソフトウェアの脆弱性の管理手法の在り方についてユースケースを交えた検討を実施している。そのような状況等を踏まえ、経済産業省では、CPSF に基づく具体的なソフトウェアのセキュリティ対策手法等の検討を行うため、「サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース」（以下、「ソフトウェアタスクフォース」という。）¹を 2019 年 9 月に設置した。

ソフトウェアタスクフォースの議論においては、産業界における OSS 利活用の重要性が高まる一方で、企業が OSS を含むソフトウェアの管理手法、脆弱性対応等に課題を抱えており、それらの課題に対し各企業において様々な対応をしている現状を踏まえ、産業界での知見の共有が有効であるとの認識に至った。

本事例集は、企業の OSS 利活用に関する課題の観点を整理し、その観点ごとに各種事例を取りまとめ、各企業が自社の「OSS の利活用及びそのセキュリティ確保に向けた管理手法」を検討する際の参考情報を提供することで、OSS の留意点を考慮した適切な OSS 利活用を促進することを目的としている。また、企業の OSS 利活用の障壁を取り除くことでより一層の OSS 利活用を促し、産業界において OSS のメリットを享受することで競争力向上につながることを期待するものである。

¹ サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース

https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunya_odan/software/index.html

2. OSS の概要

2.1. OSS とは

OSS とは、ソフトウェアのソースコードが公開され、利用や改変、再配布を行うことが誰に対しても許可されているソフトウェアのことである。OSS の概念及び志向は、世界中のユーザーがソースコードを共有の知的財産として扱い、修正や改良を重ねながらより良いソフトウェアへと磨き上げていくことである。

一般的な OSS の条件として、オープンソースに関する文化の啓発活動を行っている Open Source Initiative²が策定、公開している「オープンソースの定義」(The Open Source Definition) と呼ばれるライセンス文書³が広く知られている。

「オープンソースの定義」には、OSS が満たすべき条件として、以下があげられている。

1. 自由な再配布を認めること
2. ソースコードの利用の自由を認めること
3. 派生物の利用の自由を認めること
4. 原作者のソースコードの完全性を示すこと
5. 特定人物・集団に対する差別を禁止すること
6. 使用分野に対する差別を禁止すること
7. ライセンスの権利配分を認めること
8. 特定製品への限定適用を禁止すること
9. OSS と共に配布される他のソフトウェアを制限してはならないこと
10. 技術中立的であること

さらに、OSS の大きな特徴として、OSS コミュニティ（以下、「コミュニティ」という。）による開発や改善、ノウハウの共有等の活動があげられる。コミュニティは、特定の OSS の利用者、開発者、愛好者らによって構

² 1998 年に米国で設立された「オープンソース」の利益を啓発、支持し、オープンソースコミュニティの様々な人達との橋渡しを目的とする団体

<https://opensource.org/>

³ The Open Source Definition

<https://opensource.org/docs/osd>

成され、非営利目的で運営されている。世界中のメンバー間でソースコードを共有し、共同で開発に携わったり、メーリングリストで関連情報を発信したり、勉強会を開催したりする等、OSS の利活用を促進するエコシステムにおいて大きな役割を担っている。企業や組織にとって、コミュニティ活動が活発なことで高品質な OSS を利用できるようになることは大きなメリットとなるため、OSS の利活用が進む中、社員のコミュニティ活動を積極的に支援する企業や組織も増えている。主なコミュニティとして、Linux Foundation や Apache Software Foundation があげられる。

なお、OSS に対比するソフトウェアの捉え方として、プロプライエタリソフトウェアがあげられる。これは、ソフトウェア配布者がその知的財産を保持しており、改変や複製が制限されているソフトウェアである。具体的にはソースコードを非公開にしたり、ソフトウェア使用許諾契約によって法的にその使用を制限したりするものを指している。主には、有償の商用ソフトウェアとして市場に流通しているものの、無償で提供されるソフトウェア（フリーソフトウェア）も存在する。

2.2. OSS の利活用領域

OSS の利活用領域は、OS⁴から業務アプリケーション、データベース、ビッグデータ、AI（人工知能）等、多岐に渡る。主要な OSS を理解するために、日本 OSS 推進フォーラム⁵が取りまとめている OSS 鳥瞰図を参照することが有効である。これは、様々な領域で用いられている OSS を俯瞰的に把握できるように図示したものである（図 2.2-1）。

OSS鳥瞰図 2020年版 (2020/06/30)

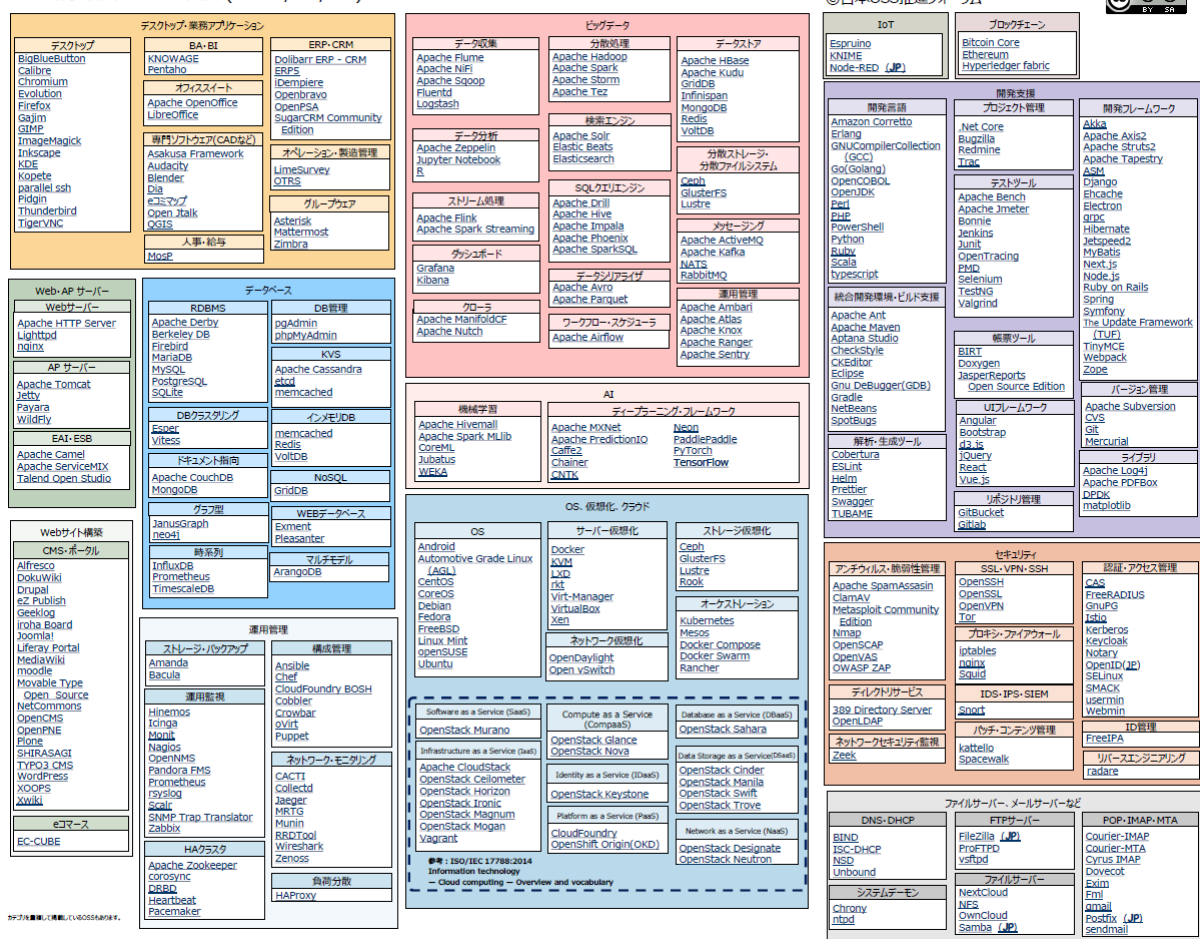


図 2.2-1 OSS 鳥瞰図 2020 年度版⁶

⁴ Operating System

⁵ 日本 OSS 推進フォーラム

<http://ossforum.jp/>

⁶ OSS 鳥瞰図 (2020 年版)

例えば、OS の領域においては、スーパーコンピュータ、サーバ、パーソナルコンピュータ、組み込みシステム（スマートフォンや家電製品等）等、幅広い種類のハードウェアに OSS である Linux が搭載されている。Linux は最も成功した OSS プロジェクトの一つであり、その広範な利活用範囲から、生活を支える製品やサービスに欠かすことのできないソフトウェアと言える。これは、前述のとおり、「無償である」「利用や改変、再配布可能である」等の特徴により、システムや製品を開発する企業に多くの便益があるために OSS が広く受け入れられていることを示している。また、その便益は、システムや製品の最終的なユーザーである企業や一般消費者にも価格面等において還元されている。

2.3. OSS 利活用のメリット

OSS を利活用するメリットは多岐に渡る。主なメリットとしては、以下の点があげられる。

- ① 開発の効率化による開発費の抑制、開発期間短縮
- ② 高い安定性、品質、透明性の確保
- ③ 豊富な種類による新たな価値創出、ベンダロックインの回避

以下、各メリットの詳細について記載する。

① 開発の効率化による開発費の抑制、開発期間短縮

開発者が必要とする機能が既に OSS に実装されている場合、それを活用することで開発費用の抑制や開発期間の短縮が期待できる。競争環境、事業環境の変化の激しい ICT 関連産業においては、開発を効率化することで非常に大きなアドバンテージを得ることが可能となる。

② 高い安定性、品質、透明性の確保

OSS はオープンソースの志向に基づき、不特定多数の利用者による改良が重ねられており、ソフトウェアとしての高い安定性、品質を有している。多数の利用者による利用実績の積み重ねもあるため、開発者としては安心して使用することができる。また、ソースコードが公開されていることもあり、不正なプログラムや脆弱性などを常に確認できるといった透明性も持ち合わせている。

③ 豊富な種類による新たな価値創出、ベンダロックインの回避

現在では OSS の種類が非常に豊富になっており、多くの機能が OSS で利用可能となっている。AI 等を用いた最新の技術が実装される等、OSS を活用することで、これまでにない新しい価値を創出したり、既存業務を大幅に効率化できる可能性がある。また、ベンダロックイン⁷を回避できるといった側面も持ち合わせている。

以上のメリットから、多くの企業が OSS を採用したシステム、製品の開発を行っている。今後もこの傾向はますます拡大していくものと考えられている。

⁷ 特定ベンダ（メーカー）の独自技術に大きく依存した製品、サービス、システム等を採用することにより、他ベンダの提供する同種の製品、サービス、システム等への乗り換えが困難になる事象

2.4. OSS 利活用の留意点

OSS の利活用には、多くのメリットがある反面、留意しなければならない点も複数存在する。主な留意点としては、以下があげられる。

- ① ライセンスコンプライアンス
- ② ライフサイクル（サポート期間等）が短い、サポートが十分ではない場合がある
- ③ サプライチェーンにおける OSS の使用

以下、各留意点の詳細について記載する。

① ライセンスコンプライアンス

OSS は、ソフトウェアごとに個別にライセンスが宣言されているため、利用者は、そのライセンスに準拠した利用が求められる。OSS 使用者は、開発担当者が多く、ライセンス等の法務の知識が十分ではない場合もある。そのような場合、気が付かないうちに OSS ライセンスに違反していた等の事象が発生する可能性があり、企業コンプライアンス上のリスクが発生していると言える。

② ライフサイクル（サポート期間等）が短い、サポートが十分ではない場合がある

OSS は、商用ソフトウェアと比較して、ソフトウェアとしてのライフサイクル（サポート期間等）が比較的短く、またサポートが十分ではない場合がある。このため、バグやソフトウェア脆弱性等が判明した場合にユーザー側で対応しなければならない等、利活用にあって一定のスキル・ノウハウが必要となる。

③ サプライチェーンにおける OSS の使用

OSS は、自社における使用のみならず、自社の製品やシステム開発のサプライチェーンを構成する企業等において使用されている可能性がある。そのような場合でも、納品物内の使用 OSS を把握し、前述のライセンス対応やバグ・ソフトウェア脆弱性等への対応を自社における OSS 使用時と同じように実施しなければならない。そのため、サプライチェーン各社の使用 OSS に関する情報を適切に吸い上げる必要がある。

これらの留意点に適切に対応することが OSS 利活用を行う上で大変重要になる。なお、これらの留意点に適切に対応する際に発生する課題については、その観点を 3.2 章において整理する。

2.5. OSS に関わる係争及びインシデント事例

2.4 章で述べた留意点を補足する観点で、過去の OSS に関わる係争、及びインシデント事例を以下に示す。

① OSS ライセンスに関わる係争事例 ―航空機内エンターテインメントソフトウェアの例⁸―

2017 年 3 月、某社が開発した航空機内エンターテインメントソフトウェアについて、競合する企業からライセンス違反があるとしてニューヨーク連邦裁判所に提訴された。提訴した企業は、Linux ベースのソフトウェアのソースコードが適切に開示されていないことを指摘し、本件は過失によるライセンス違反ではなく、競合企業が類似のソフトウェアを開発するのを妨害する目的で故意に実施されているとして、1 億ドルの賠償を要求した。2018 年 1 月に両社は和解したものの、賠償額は明かされていない。

② OSS の脆弱性に関わるインシデント事例 ―Heartbleed⁹の例⁸―

2014 年 4 月、OpenSSL (SSL/TLS¹⁰プロトコルの OSS ライブラリ) の Heartbeat 機能において脆弱性が見つかったことが公表された。Heartbeat 機能は、機器間で通信が行われていない間も TLS セッションの接続を維持し、通信相手が存在しているかを確認する機能である。脆弱性を含んだバージョンの OpenSSL サーバ宛に、細工をした Heartbeat のリクエストを送ることで、その返答の際にサーバのメモリ上のデータが含まれ、ID/パスワードや SSL サーバ証明書の秘密鍵が漏えいする可能性があった (図 2.5-1)。

⁸ サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース 第 1 回資料
https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/pdf/001_04_00.pdf

⁹ この脆弱性は「Heartbeat」機能に由来することから「Heartbleed (ハートブリード、心臓出血)」と命名された。

¹⁰ SSL : Secure Sockets Layer TLS : Transport Layer Security
いずれもインターネット上でデータを暗号化して送受信するためのプロトコルの一種である。

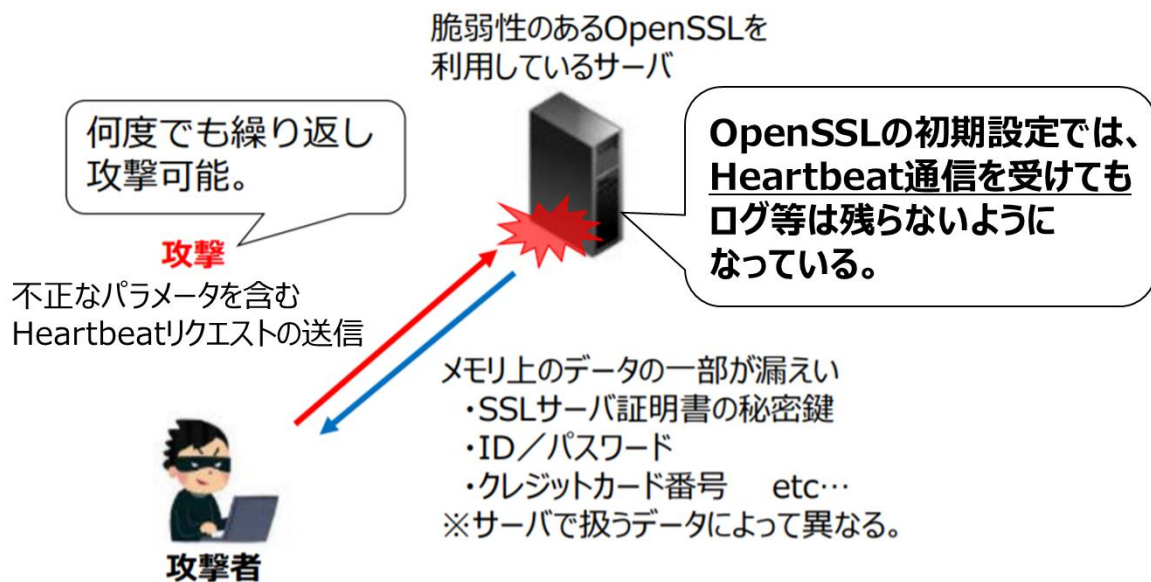


図 2.5-1 Heartbeat 機能を用いた攻撃イメージ¹¹

¹¹ サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース 第 1 回資料より
抜粋

[https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/
software/pdf/001_04_00.pdf](https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/pdf/001_04_00.pdf)

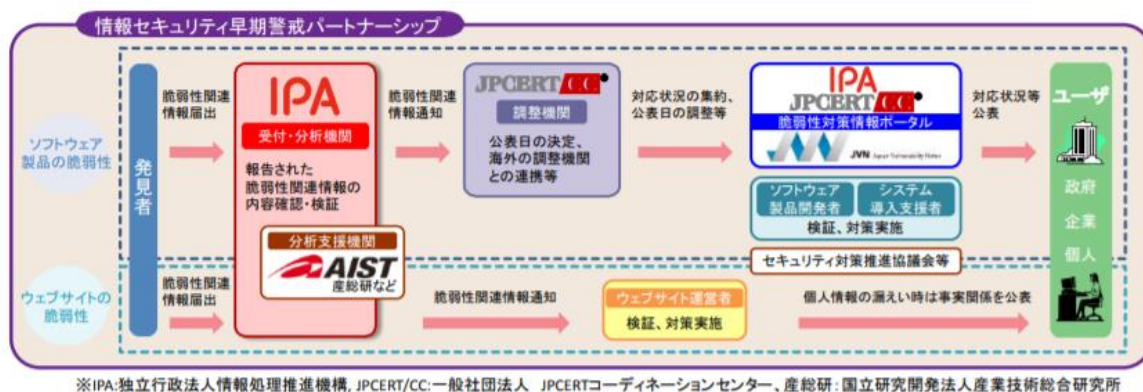
2.6. OSS 利活用等に寄与する主な取組

2.6.1. 情報セキュリティ早期警戒パートナーシップ・Japan Vulnerability Notes

OSS の利活用を支援する主な取組として、情報処理推進機構（IPA）及び JPCERT コーディネーションセンター（JPCERT/CC）によって運営されている情報セキュリティ早期警戒パートナーシップ、及び Japan Vulnerability Notes（JVN）があげられる。製品・サービスやシステムに OSS を利活用しており、その OSS に関わる脆弱性が判明した場合、その脆弱性に迅速かつ適切に対応することは、セキュリティを保つ上で大変重要となる。情報システム早期警戒パートナーシップ及び JVN は、これらの一連の対応を行うにあたり必要な情報をユーザーに提供している。

情報セキュリティ早期警戒パートナーシップは、2017 年の経済産業省告示「ソフトウェア製品等の脆弱性関連情報に関する取扱規程」に基づき、国内におけるソフトウェア等の脆弱性関連情報を適切に流通させることを目的に作られている枠組みである。IPA が一般人や研究者が発見したウェブサイト及びソフトウェア製品に関するセキュリティ上の問題（脆弱性）を受け付け、JPCERT/CC が脆弱性の発見者、ソフトウェア製品開発者、ウェブサイト運営者等と協力・連絡調整をしながらその対処を取りまとめている（図 2.6-1）。

JVN は、情報セキュリティ早期警戒パートナーシップに基づき発見された脆弱性情報等を、取りまとめられた対処法等と共に公開する情報ポータルサイトである。情報セキュリティ早期警戒パートナーシップに基づく情報以外に、CERT/CC など海外の調整機関と連携した脆弱性情報も公表している（図 2.6-2）。具体的な公表内容としては、脆弱性が確認された製品とバージョン、脆弱性の詳細や分析結果、製品開発者によって提供された対策や関連情報へのリンク等で、対策にはパッチだけではなく回避策（ワークアラウンド）が含まれることもある。



の OSS に係る取組を集めたガイド「企業のためのオープンソースガイド」が公開されている（日本語訳版有）。「企業のためのオープンソースガイド」は、「オープンソースプログラムオフィスの運用」と「組織におけるオープンソースプロジェクトの管理」の 2 つのテーマに基づき、以下の 12 のガイドで構成されている¹⁴。

<オープンソースプログラムオフィスの運用>

- オープンソースプログラムの作成
- オープンソース管理ツール
- オープンソースプログラムの成功度を測る
- オープンソースデベロッパーの採用
- オープンソースコミュニティへの参加
- オープンソースコードの使用

<組織におけるオープンソース プロジェクトの管理>

- オープンソースプロジェクトを立ち上げる
- オープンソース開発の効果を高める
- オープンソースガイド推奨図書
- オープンソースプロジェクトを終了させる
- オープンソースコミュニティでリーダーシップを構築する
- オープンソース戦略の策定

Linux Foundation では、この他にも OSS 利活用戦略を立案するための実践的アプローチを解説した「エンタープライズオープンソース：実践的入門」¹⁵や、各業界における OSS の先進活用例を紹介する「ソフ

¹⁴ 企業のためのオープンソースガイド

<https://www.linuxfoundation.jp/resources/open-source-guides/>

¹⁵ エンタープライズオープンソース：実践的入門

<https://www.linuxfoundation.jp/open-source-management/2018/08/enterprise-open-source-practical-introduction/>

トウェア定義が進む各業界：オープンソースによる変革」¹⁶等、有用なガイド、ホワイトペーパーを公開している。

¹⁶ ソフトウェア定義が進む各業界：オープンソースによる変革

<https://www.linuxfoundation.jp/blog/2020/09/software-defined-vertical-industries-transformation-through-open-source/>

3. 事例の整理方法

3.1. OSS を取り巻く商流・ステークホルダの整理

OSS の商流（サプライチェーン）は、多種多様なステークホルダにより形成されている（図 3.1-1）。OSS は、OSS 開発コミュニティやそのコミュニティ参加企業といった開発ステークホルダによる OSS の開発・提供からその商流が始まる。その後、OSS を商用パッケージとして流通させるディストリビュータ、OSS を用いて製品やシステムを開発する最終製品メーカー、サプライヤ、SIer¹⁷、OSS を用いたクラウドサービス等を提供するサービス事業者等の中間層を介する。最終的に、製品であれば製品のエンドユーザーとしての企業・消費者、システムであればエンドユーザーが製品やシステムに対価を支払い、その便益を享受する。

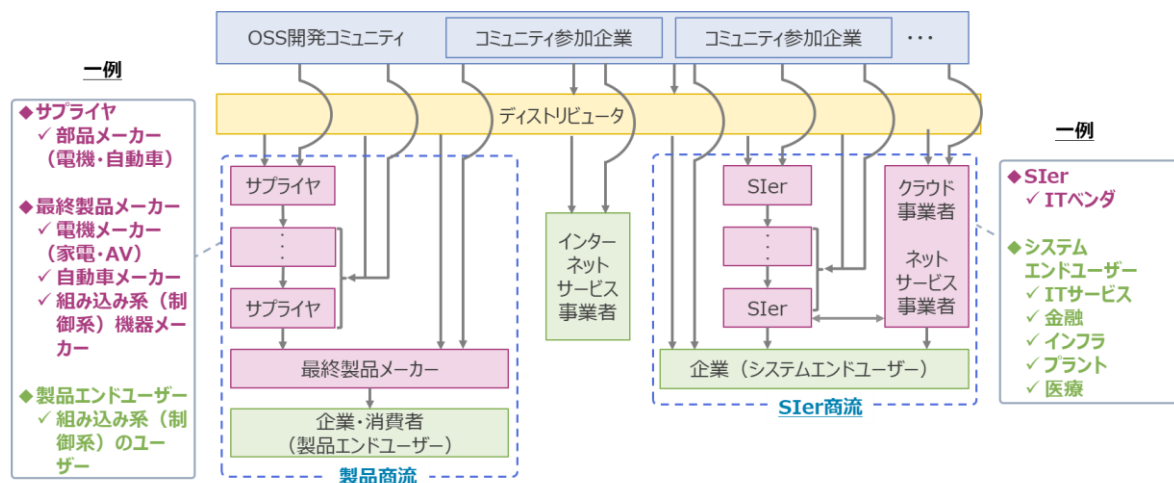


図 3.1-1 OSS の商流とステークホルダ¹⁸

OSS の商流は、主に最終製品を供給するために発生する商流（製品商流）と企業におけるシステム導入に伴い発生する商流（SIer 商流）に大別することが可能である。本事例集では、OSS を実際に利用する主体となる、ディストリビュータ以降の製品商流、SIer 商流におけるステークホルダの取組を紹介する。それぞれの商流の概要、及び主なステークホルダは以下のとおりである。

- 製品商流

最終製品メーカーと最終製品メーカーに部品供給するサプライヤによって形成される商流である。製品は、

¹⁷ システムインテグレータ（Systems Integrator）

¹⁸ ヒアリング等を基に作成

最終的に製品エンドユーザーが購入する。サプライヤは、サプライチェーンを形成しており、重層的に部品供給を受けている場合がある。最終製品メーカーの製品開発、及びサプライチェーン各段階のサプライヤにおける部品開発において広く OSS が利活用されている。製品商流の主なステークホルダは以下のとおりである。

➤ サプライヤ :

最終製品メーカーに部品供給するメーカーである。OSS 商流においては、OSS 等を含むソフトウェアを組み込んだ部品（製品）を製造し、納品するメーカーが対象となる。最終製品によって、固有の、もしくは専門性を持つサプライヤが存在する（自動車部品メーカー、電子部品メーカー等）。

➤ 最終製品メーカー :

自社で開発した部品、及びサプライチェーン（サプライヤ）から供給された部品を基に最終製品を製造するメーカーである（自動車メーカー、電機メーカー等）。自社開発において OSS を使用する場合もあれば、サプライヤから供給された部品に OSS が使用されている場合もある。

➤ 製品エンドユーザー :

最終製品メーカーが製造した製品を購入、使用する組織、個人は全て製品エンドユーザーである。

● SIer 商流

システムエンドユーザー（企業等）からの発注により、SIer が開発を行い、業務システム等を納品する商流である。SIer はサプライチェーンを形成しており、重層的な業務委託により開発を実施している場合がある。あわせて、クラウドサービス等のサービスも組み合わせつつ、最終的にシステムを構築している。サプライチェーン各段階の SIer、及び各サービス事業者において広く OSS が利活用されている。SIer 商流の主なステークホルダは以下のとおりである。

➤ サービス事業者 :

クラウドサービスやネットサービス等を SIer やシステムエンドユーザーに提供している事業者があげられる。SIer やシステムエンドユーザーはそれらのサービスを活用してシステム構築を行っている。

➤ SIer :

情報システムのインテグレーションを行う事業者や、情報システムの開発・構築を受託し納品する事業者等があげられる。自社の IT 製品を持ち、その提供を行いつつ開発を受託する事業者もいる。業界ごとに、使用する業務システムの種類や必要となる特性に共通点があるため、業界ごとに固有の、もしくは専門性を持つ SIer やシステム開発事業者が存在する（金融業向けの SIer、製造業向けの SIer 等）。

➤ システムエンドユーザー :

企業や官公庁、自治体等、業務システムを使用する組織は全てシステムエンドユーザーである。

● インターネットサービス事業者

インターネットサービスを提供する事業者である。インターネットサービス事業者は、OSS を活用してサービスを提供しており、BtoB または BtoC においてサブスクリプション型、ライセンス販売や無償等による提供を行っている。

3.2. OSS の利活用に係る課題観点の抽出

3.2.1. サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォースにおける課題の整理

ソフトウェアタスクフォースでは、過去の国内外におけるソフトウェアにおける管理、脆弱性対応、ライセンス等に関する事例を基に、OSS を含むソフトウェアの利活用及びそのセキュリティ確保に向けた管理手法に係る課題を「検討の方向性」として、図 3.2-1 のとおり整理を行っている。

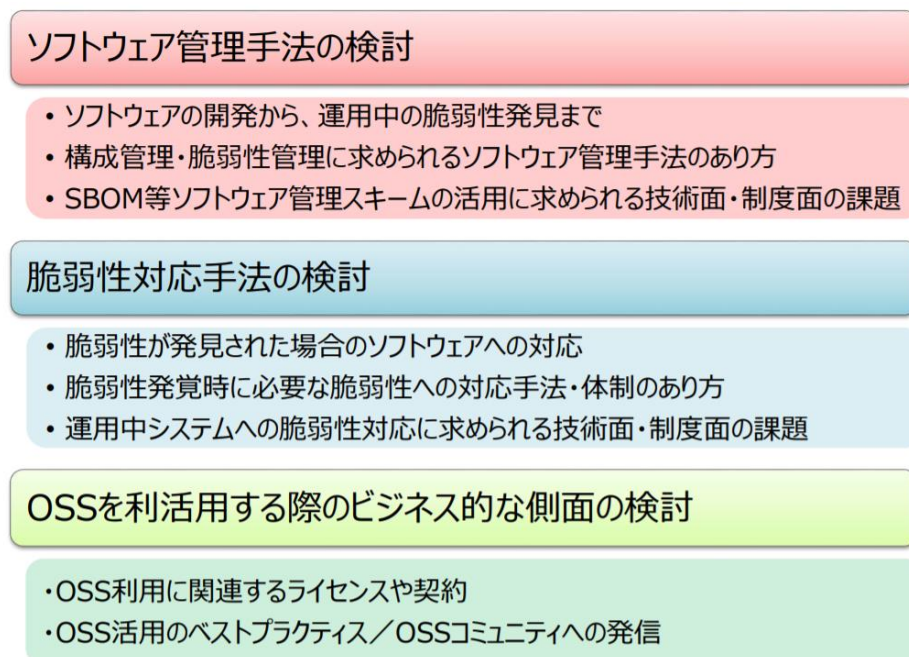


図 3.2-1 タスクフォースにおける検討の方向性¹⁹

ソフトウェアタスクフォースにおいては、OSS の脆弱性対応、ライセンス対応において、適切な OSS 管理を行う重要性を認識した。また、OSS を管理する粒度や、組織としてどのように取り組むかといった点等を今後の課題としてあげ、OSS 管理に向けた制度面、体制面等の整備の推進を図る必要性があるとした。これら課題への対応として、各社の OSS 管理に係る取組の知見共有が重要であるとし、OSS の利活用及びそ

¹⁹ サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース 第3回資料より抜粋
https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunya_odan/software/pdf/003_03_00.pdf

のセキュリティ確保に向けた管理手法に係る事例集等の作成が有効であるという共通認識を得ている。

3.2.2. OSS 利活用に係る課題の観点

OSS の利活用及びそのセキュリティ確保に向けた管理手法に係る課題の観点について、ソフトウェアタスクフォースの議論内容やタスクフォース委員（有識者・企業）へのヒアリングを基に整理を行った。

本事例集では、OSS ステークホルダが OSS を利活用する際に抱える主な課題を以下の観点で分類している。

- 選定評価（Selection Evaluation）
- ライセンス（Licensing）
- 脆弱性対応（Vulnerability Response）
- 保守、品質保証（Maintenance and Quality Assurance）
- サプライチェーン管理（Supply Chain Management）
- 個の能力、教育（Personal Competence and Education）
- 組織体制（Organizational Structure）
- コミュニティ活動（Community Activities）

上記の観点ごとに、その主な課題を整理した表を以下（図 3.2-2）に示す。なお、各観点は、OSS を利活用する際、選定や管理等のプロセスにおいて発生する課題の観点（「プロセス別観点」）、OSS に携わる人材や組織のリテラシー向上、体制整備において発生する課題の観点（「人・組織に関する観点」）、OSS コミュニティ活動等、OSS のエコシステムへの貢献・拡大に対する課題（「コミュニティに関する観点」）に分類している。



図 3.2-2 OSSに係る観点とその課題（一例）²⁰

²⁰ ヒアリング等を基に作成

3.3. 紹介事例一覧

本事例集は、ヒアリングにより収集した国内企業の事例（20 事例）と、公開文献の調査により収集した海外事例（3 事例）を掲載している。

3.1 章における OSS を取り巻く商流・ステークホルダの整理、及び 3.2 章における OSS 課題観点の整理を基に、本事例集における各事例の対応表を以下（表 3.3-1）に示す。各事例の詳細については、次章以降にて記載する。表 3.3-1 には、“L1”、“S1”のように各事例に関連する観点を付与しており、該当する観点の取組の説明箇所へ遷移することが可能としている。なお、表 3.3-1 は本事例集で紹介している取組の観点を一覧として整理したものであり、各社の取組のすべてを網羅的に示したのではない。

表 3.3-1 掲載事例一覧

商流	ステークホルダ	事例企業	事例に関連する観点							
			選定評価 (S)	ライセンス (L)	脆弱性対応 (V)	保守・品質保証 (M)	サプライチェーン管理 (S)	個の能力・教育 (P)	組織体制 (O)	コミュニティ活動 (C)
ヒアリング実施企業										
製品商流	最終製品メーカー	トヨタ自動車		L1			S1			C1
		ソニー		L2-1 L2-2	V2		S2		O2	C2
		オリンパス		L3			S3	P3	O3	C3
		日立製作所		L4						C4
		オムロン			V5-1 V5-2		S5	P5	O5-1 O5-2	
		東芝	S6	L6	V6	M6	S6	P6	O6	C6
		三菱電機インフォメーションシステムズ (MDIS)	S16	L16	V16	M16				C16-1 C16-2
	サプライヤ	デンソー		L7	V7		S7			

商流	ステークホルダ	事例企業	事例に関連する観点							
			選定評価 (S)	ライセンス (L)	脆弱性対応 (V)	保守・品質保証 (M)	サプライチェーン管理 (S)	個の能力・教育 (P)	組織体制 (O)	コミュニティ活動 (C)
製品商流 & SIer 商流	SIer& 最終製品メーカー	富士通	S8	L8	V8-1 V8-2	M8	S8	P8	O8-1 O8-2	C8
		日本電気 (NEC)	S9-1 S9-2	L9-1 L9-2	V9-1 V9-2	M9		P9	O9-1 O9-2	
SIer 商流	SIer	日本電信電話 (NTT)			V10				O10	
		匿名企業 A 社	S11			M11				C11
	SIer& エンドユーザー	匿名企業 B 社			V12			P12		
	エンドユーザー	損害保険 ジャパン		L13	V13			P13		
		Visional グループ			V14					
		サイボウズ	S15	L15	V15			P15	O15	C15
		OSSTech		L17	V17		S17			
	SCSK	S20		V20-1 V20-2		S20	P20	O20		
インターネットサービス事業者	ヤフー	S18	L18	V18	M18	S18	P18	O18	C18	
	ラキール	S19	L19	V19				O19		
文献調査実施企業等										
SIer 商流	サービス事業者	マイクロソフト			V20					
	エンドユーザー	ザランド		L21						

商流	ステークホルダ	事例企業	事例に関連する観点							
			選定評価 (S)	ライセンス (L)	脆弱性対応 (V)	保守・品質保証 (M)	サプライチェーン管理 (S)	個の能力・教育 (E)	組織体制 (O)	コミュニティ活動 (C)
その他（調査結果）		Census II プロジェクト			V22					

4. 事例（ヒアリング調査）

4.1. トヨタ自動車 ～サプライチェーンにおけるソフトウェア使用状況把握～

■ 企業情報・本事例集における位置づけ

本社	愛知県豊田市
業種	製造（自動車）
従業員数（連結）	370,870 人（2019 年 3 月末現在）
商流	製品商流/最終製品メーカー
観点	「ライセンス」「サプライチェーン管理」「コミュニティ活動」

■ 本事例のエッセンス

- ソフトウェア管理ルール明確化の取組により、ライセンス違反のリスク低減及びサプライチェーン全体における OSS に係るセキュリティ脆弱性への対応力向上に努めている。
- 使用ソフトウェアに関するガイドライン締結においては、サプライヤに対して、OSS コンプライアンスの重要性を強調することで理解を求めた。
- サプライヤからの使用ソフトウェアリスト提供にあたっては、運用面を考慮し、SPDX Lite のフォーマットを採用した。
- OSS 管理に係る社内の取組を、コミュニティ活動において共有・議論することで、オープンな PDCA サイクルを形成し、サプライチェーン全体における OSS コンプライアンスの底上げに貢献している。

■ 背景・課題

自動車業界は、最終製品メーカーのみならず、重層的なサプライヤが存在し、多数の企業がサプライチェーンを形成している。サプライチェーン各企業においては、OSS が使用されており、OSS ライセンスを遵守するために、その使用状況を正確に把握することが求められる。しかしながら、サプライチェーン各社において、使用ソフトウェアの管理粒度、ガバナンス体制は異なっているため、それらをいかに統一的な運用に落とし込むかが課題となっている。

■ 取組

<【ライセンス(L1)】【サプライチェーン管理(S1)】サプライチェーンにおけるソフトウェア（OSS 含む）使用状況把握>

トヨタ自動車では、OSSコンプライアンスに係る運用ルール、監査ルール、帳票等の社内プロセス、権限、組織体制等の整備を全社直轄組織の下で取り組んでいる。特に、サプライチェーンにおける使用ソフトウェアの把握については、サプライヤ（Tier 1²¹）に対して、以下の 2 点の取組を実施している（表 4.1-1）。

- ① 各サプライヤと使用ソフトウェアに関するガイドラインを合意
- ② 部品供給に際して、自社と各サプライヤのエンジニア部門同士で使用ソフトウェア報告の運用規則を合意

表 4.1-1 ガイドライン、運用規則における規定項目²²

ガイドライン (会社間のベーシックな ルールの合意)	● サプライチェーンにおけるOSSコンプライアンスの役割 ● 責任分界 ● 各ライセンスの取り扱い方法 ● 使用ソフトウェアリスト提供の合意
運用規則 (実務レベルの 運用ルールの合意)	● リストのフォーマット ● ワークフロー ● 報告のタイミング ● 連絡先

以下、各取組の詳細について記載する。

- ① 各サプライヤと使用ソフトウェアに関するガイドラインを合意

サプライチェーンにおける OSS コンプライアンスの役割をはじめ、使用ソフトウェアリスト提供をすることや注意を払うべきライセンス等について会社間でガイドラインの合意を行っている。それまでの自動車業界では、供給先に対して必要以上の情報提供を避ける商習慣が一部存在していたものの、ガイドラ

²¹ 最終製品メーカーに直接部品を供給するメーカー

²² Open Compliance Summit 2019 の資料を基に作成

https://static.sched.com/hosted_files/ocs19/d3/Open%20Source%20Compliance%20in%20Supply%20Chains.pdf

インの合意に向けて、特にライセンス面におけるコンプライアンスの重要性を強調し、各サプライヤーへ理解を求めた。

- ② 部品供給に際して、自社と各サプライヤーのエンジニア部門同士で使用ソフトウェア報告の運用規則を合意

前述のガイドラインに基づき、各サプライヤーとの供給関係（プロジェクト）ごとに実務（エンジニア）レベルで、供給部品における使用ソフトウェアリスト提供に関する運用規則の合意を行っている。具体的には、規則の適用範囲（製品、サービス等）、使用ソフトウェアリストの報告方法（誰から誰へ報告するか、報告のタイミング（開発段階の報告含む）等）、フォーマット、ライセンスで要求されている著作権情報、ソースコード開示の具体的な対応方法（マニュアルへの掲載、Web での公開等）等をプロジェクトごとに定めている。

各サプライヤーにおけるハンドリングのしやすさを考慮した結果、使用ソフトウェアリストのフォーマットは OpenChain²³で議論されている SPDX Lite を採用している。SPDX Lite は、SPDX²⁴の簡易バージョンであり、Excel 等での管理も可能である（図 4.1-1）。SPDX Lite においてリスト化する情報や記載ルールは、最善の運用を目指し、継続的に改善が検討されている。

SPDX Liteの項目例			-パッケージダウンロード位置（入手先）			-ライセンスへのコメント		
-パッケージ名			-解析したファイル（手作業の場合false）			-著作権テキスト		
-パッケージSPDX識別子			-ホームページ（開発コミュニティサイト）			-改変の有無		
-パッケージバージョン			-結論されたライセンス			-パッケージに関するコメント		
-パッケージファイル名			-宣言されたライセンス			-ライセンス識別子		

SPDXサンプルの一部抜粋								
3.1	3.2	3.3	3.4	3.7	3.8	3.11	3.13	3.15
Package Name	Package SPDX Identifier	Package Version	Package File Name	Package Download Location	Files Analyzed	Package Home Page	Concluded License	Declared License
項	パッケージ名	パッケージSPDX識別子	パッケージファイル名	パッケージダウンロード位置（入手先）	解析したファイル（手作業の場合false）	ホームページ（OSS開発コミュニティサイト）	結論されたライセンス	宣言されたライセンス
	LibXML2		2.99	libxml2-2.9.9.tar.gz	http://xmlsoft.org/libxml2/	FALSE	http://xmlsoft.org/	MIT
	PCRE		8.4343	pcre-8.43.tar.gz	http://the.pcre.org/pub/pcre/	FALSE	https://www.pcre.org/	BSD-3-Clause
	SQLite		3300100.1	sqlite-autoconf-3300100.tar.gz	https://www.sqlite.org/download.html	FALSE	https://www.sqlite.org/index.html	その他(ライセンス名を記載)
	Zlib (1.2.3)		1.2.11	zlib-1.2.11.tar.gz	https://www.zlib.net/	FALSE	https://www.zlib.net/	Zlib
	cURL (7.41.0)		7.66.0	curl-7.66.0.tar.bz2	https://github.com/curl/curl/releases	FALSE	https://curl.haxx.se/	MIT

図 4.1-1 SPDX Lite フォーマット（サンプル）²⁵

²³ Linux Foundation のプロジェクト。サプライチェーン全体にわたって OSS コンプライアンスを実現するための業界標準の作成、普及を目的としている。

<https://www.openchainproject.org/ja/>

²⁴ ソフトウェアパッケージデータ交換形式（SPDX : Software Package Data Exchange）は、Linux Foundation が支援するソフトウェアパッケージに関連するコンポーネントやライセンス、コピーライト等の情報をやり取りするための標準的なフォーマットである。

²⁵ サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース 第3回資料より抜粋

以上のような取組により、ソフトウェア管理に対する重要性の認識が高まったとともに、その管理ルールが明確となり、サプライヤごとのソフトウェア管理品質のバラつき低減が可能となった。これによりライセンス違反のリスクが低減するのみならず、サプライチェーン全体における OSS に係るセキュリティ脆弱性への対応力が高まっている。

<【コミュニティ活動(c1)】OSS 利活用促進に向けたコミュニティ等における活動>

トヨタ自動車では、サプライチェーンにおけるソフトウェア使用状況把握の取組と並行して、それらの取組に関する標準化に貢献している。一例として、OpenChain の活動があげられる。

OpenChain は、サプライチェーン全体にわたって OSS コンプライアンスを実現する業界標準の作成・普及を目的とした Linux Foundation のプロジェクトである。日本からは、電機、情報通信機器メーカーを中心に多くの企業が参加している。プロジェクトの主な成果物²⁶（図 4.1-2）は、以下の 3 点である。

- ① 仕様（Specification）：企業が組織内に確立すべきコンプライアンスプログラムの要件を定義
- ② 適合（Conformant）：仕様に適合しているか否かを Web アンケートで回答することで、認証を取得可能。原則、自己認証だが第三者認証もある
- ③ カリキュラム（Curriculum）：仕様への適合のために各社内で利用可能な教育資料集

https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/pdf/003_04_00.pdf

²⁶ トヨタ自動車はこれらの成果物を基に、前述の OSS コンプライアンスに係る社内体制を構築している。

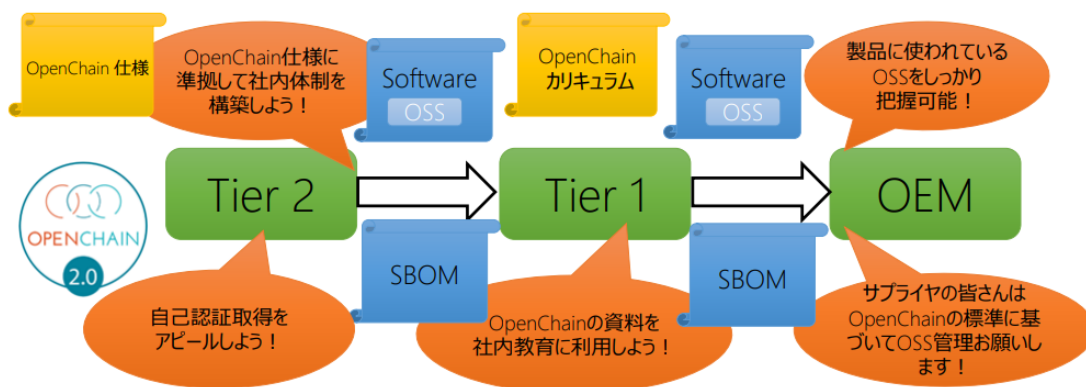


図 4.1-2 OpenChain の成果物の活用イメージ²⁷

①の仕様については、2020 年 12 月に ISO/IEC5230 として国際標準化された。国際標準化により認知度が向上することで、より多くの企業が仕様への適合について認証を取得することが期待されている。トヨタ自動車では、自社の OSS 管理の取組について、国際標準化の前から OpenChain 仕様に合わせたチェックと不足部分の対応を進めており、国際標準化後、世界で初めて仕様への適合（自己認証）を発表している。現在、仕様に適合している部署が一部に限られているため、全社に広めるべく、取組が進められている。

OpenChain には、複数のワーキンググループ（WG）があり、トヨタ自動車は、Japan WG²⁸や Automotive WG²⁹において中心的な役割を担い、貢献している（表 4.1-2）。Japan WG では SPDX Lite の議論をはじめ、7 つのサブワーキンググループ（SWG）がそれぞれのテーマの下に活動している。Automotive WG では、ベストプラクティスの共有や OSS コンプライアンスの重要性に関する周知活動等を実施している。

²⁷ サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース 第 3 回資料より抜粋

https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/pdf/003_04_00.pdf

トヨタ自動車では図中の SBOM（Software Bill of materials）に SPDX Lite を採用している。

²⁸ OpenChain に参加する日本企業が主体となっている WG である。

²⁹ OpenChain に参加する自動車関連企業が主体となっている WG である。

表 4.1-2 各 WG の概要³⁰

WG 名称	OpenChain Japan WG	OpenChain Automotive WG
設立時期	2017 年 12 月	2019 年 7 月
参加企業	トヨタ自動車、ソニー、日立等、 国内 70 社以上	トヨタ自動車、ボッシュ、パナソニック等 (国内外 12 の自動車メーカー含む)
目的	OSS コンプライアンスに関して、日本企業ならではの課題や解決策を日本語で情報交換する場所を作ることを目的としている。	3 つの目的が設定されている。 - 自動車業界内のベストプラクティス共有 - OSS サプライチェーンマネジメントの自動車業界標準の検討 - OSS コンプライアンスの重要性の業界内での周知活動
活動内容	企業や組織がオープンソースライセンスのコンプライアンスを遵守していくための基準や支援、各情報の翻訳等を実施している。また、SWG レベルでテーマを設定し、それぞれ活動を行っている。	参加企業間で各企業の取組紹介等を実施している。2020 年度は、プラクティス集の作成等を予定している。
SWG	- Planning SWG - FAQ SWG - Leaflet to Supplier SWG - Education material for roles SWG - License information exchange SWG ※ - Tooling SWG - Promotion SWG ※SPDX Lite について議論	—

各参加企業の OSS コンプライアンスに関する取組結果は OpenChain にフィードバックされ、より良い

³⁰ サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース 第 3 回資料及びヒアリングを基に作成

https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/pdf/003_04_00.pdf

成果物を生み出すために議論される。これによって、各参加企業によるオープンな PDCA サイクルが形成されている。

また、OpenChain には、自動車業界のサプライヤ企業も参加している。したがって、最終製品メーカーのみの意図によって業界標準を策定しているわけではない。そのため、サプライヤ側からの理解も得られやすく、今後の拡大が期待されている。

トヨタ自動車は、自社の取組結果に対して、前述のオープンな PDCA サイクルを回すことによって、以降の取組の精度を高めている。あわせて、その PDCA サイクルの恩恵は、サプライチェーン全体に波及し、サプライチェーン全体における OSS コンプライアンスの取組を底上げする効果が発揮されている（図 4.1-3）。

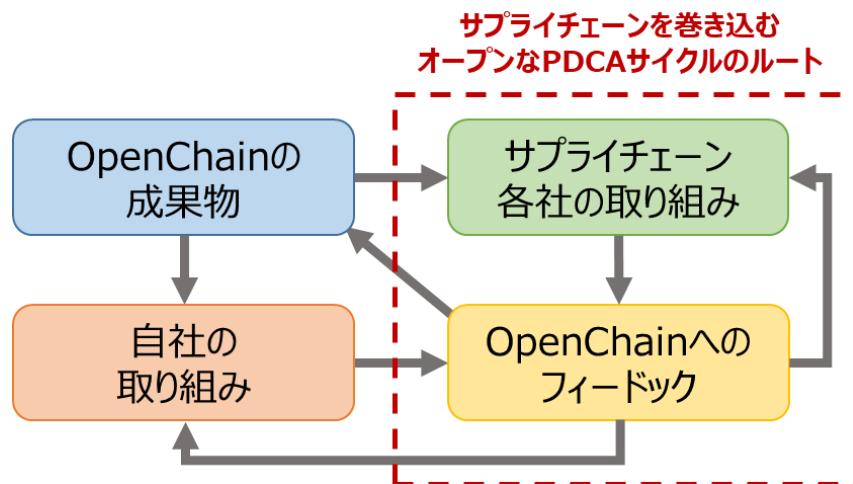


図 4.1-3 OpenChain におけるオープンな PDCA サイクル³¹

さらに、トヨタ自動車は、OpenChain の活動以外にも、OSS 利活用の促進に向けて、以下の活動を実施している。

- ① OSS ライセンス解釈の負荷を軽減するためのツールの共同開発
- ② 展示会等における SBOM の普及活動

以下、各取組の詳細について記載する。

- ① OSS ライセンス解釈の負荷を軽減するためのツールの共同開発

トヨタ自動車は、日立製作所及び日立ソリューションズと OSS ライセンス解釈の負荷を軽減するた

³¹ ヒアリングを基に作成

めのツール「OSS License Simple Viewer」を共同開発した。これは、OSS ライセンスが英語記載で複雑なものがある中、ライセンスの種類やその OSS の用途を入力するだけで、そのライセンスの責務・免責事項が表示され、簡単にその内容を把握できるツールとなっている。また、企業ごとのライセンス解釈によって、表示される責務・免責事項をカスタマイズすることも可能である。

本ツールは、より多くの企業に使用してもらい、OSS 利活用の助けとなることを目的に、ツール自体も OSS として公開³²されている。今後は、本ツールが企業における OSS 管理の仕組みやシステム等と連携しつつ、より付加価値のあるツールに改良されていくことが期待されている。

② 展示会等における SBOM の普及活動

トヨタ自動車は、OSS 利活用に伴う SBOM 作成について、展示会等でその必要性等をアピールする活動にも貢献している。一例として、2020 年 1 月に開催された CES2020³³の際に、トヨタがサポートする Linux Foundation プロジェクトである Automotive Grade Linux のブースにて、OpenChain Automotive WG の活動の一環として SPDX Lite 作成のデモを行った。無償ツールを活用して SPDX Lite を簡単に作成できること等をアピールし、来場者に SBOM 利用の啓発を図った。他にも、SBOM 作成ツールを提供している企業のセミナー等で SBOM の重要性を訴求する等の活動を行っている。トヨタ自動車としては、このような場を通じて、引き続き SBOM 管理の必要性等を含む OSS 利活用に向けた情報発信を行いたいと考えている。

³² OSS License Simple Viewer

<https://github.com/OLSV-oss/OSSLicenseSimpleViewer>

³³ 毎年 1 月に米国ラスベガスで開催される電子機器の見本市

4.2. ソニー ～各事業部による主体性のある取組～

■ 企業情報・本事例集における位置づけ

本社	東京都港区
業種	電気機器
従業員数（連結）	111,700 人（2020 年 3 月末現在）
商流	製品商流/最終製品メーカー
観点	「ライセンス」「脆弱性対応」「サプライチェーン管理」「組織体制」「コミュニティ活動」

■ 本事例のエッセンス

- OSS 管理において、一定の基準を設けることで品質は担保しつつ、各事業部の裁量に任せることで、ソニーの多様な製品群をフォローすることが可能となっている。
- OSS ライセンスコンプライアンス対応として、組織横断型の対応体制を構築するとともに、OpenChain Japan WG³⁴への参画を通じたサプライチェーン管理の強化に取り組んでいる。

■ 背景・課題

ソニーでは、多様な製品ラインナップを持つ中で、それぞれの事業部がそれぞれの製品市場に対して最適な対応を取れるよう、プロセスやルールを定めている。OSS のライセンス対応、ソフトウェア管理についても、企業として製品の品質を担保しながら、各製品の競争力を失わないよう事業部ごとに柔軟にその手法を検討・採択しなくてはならない。

■ 取組

<【ライセンス(L2-1)】【脆弱性対応(v2)】OSS 管理（製品セキュリティ）における各事業部の柔軟な対応>

ソニーは、2000 年頃から OSS 利活用のメリットを認識し、OSS を利用してきた。当初は、OSS の適切なライセンス対応に必要となる取組から実施し始め、その知見も活かしながら OSS 管理（製品セキュ

³⁴ OpenChain Japan WG の具体的な取組については、トヨタ自動車の事例に記載

リティ) にも取組を拡大してきた。

製品セキュリティの取組としては、全社の品質マネジメントシステム (QMS)³⁵に基づき、各事業部に
て各製品の使用 OSS の把握、リスト管理及び定期的な更新を徹底している。

また、OSS の脆弱性が発見された際は、Sony PSIRT³⁶が中心となり、そのハンドリング³⁷を実施する。

なお、QMS に基づく製品セキュリティの取組は、各事業部の状況に合わせて裁量を持たせる形で運用
されている。

これは、各事業部の製品によって、開発プロセスや保有するリソースが異なる中で、柔軟にその方法を
選択できることが最も効率的なためである。これにより、QMS の情報品質を担保しつつ、ソニーが持つ多
様な製品群をフォローすることが可能となっている。

<【ライセンス(L2-2)】【サプライチェーン管理(s2)】【組織体制(o2)】【コミュニティ活動(c2)】OSS ライセンスコ ンプライアンス対応に係る体制とコミュニティ活動>

ソニーでは、OSS ライセンスのコンプライアンス体制について、2010 年には組織横断型の対応組織を
構築し、現在に至っている。これは、法務、知財、広報、品質管理のコーポレート部門、ほぼすべての事
業部、海外事業拠点から選出されたメンバーが参画する社内委員会を形成し、OSS ライセンスのコンプ
ライアンスに係る取組を推進している (図 4.2-1)。これにより、社内の様々なノウハウが活用できると
ともに、取組の推進主体が各事業部に分散することで、その事業部の状況に沿った柔軟な展開が期待で
きる。

³⁵ Quality Management System: 製造物や提供されるサービスの品質を管理・監督するシステム
品質管理を中心とした組織の活動で顧客満足を達成し継続的な改善を意図する。

³⁶ Product Security Incident Response Team: 自社で製造・開発する製品やサービスを対象に、
セキュリティレベルの向上やインシデント発生時の対応を行う組織。Sony PSIRT では、各事業部と緊密に
連携し、ソニー全社の多様な製品群に対応するため、定期的に関係者とのコミュニケーションの場を設定する
等の工夫をしている。

³⁷ 製品脆弱性に関する情報の収集、受付窓口、対応判断、関係部署への伝達等を実施

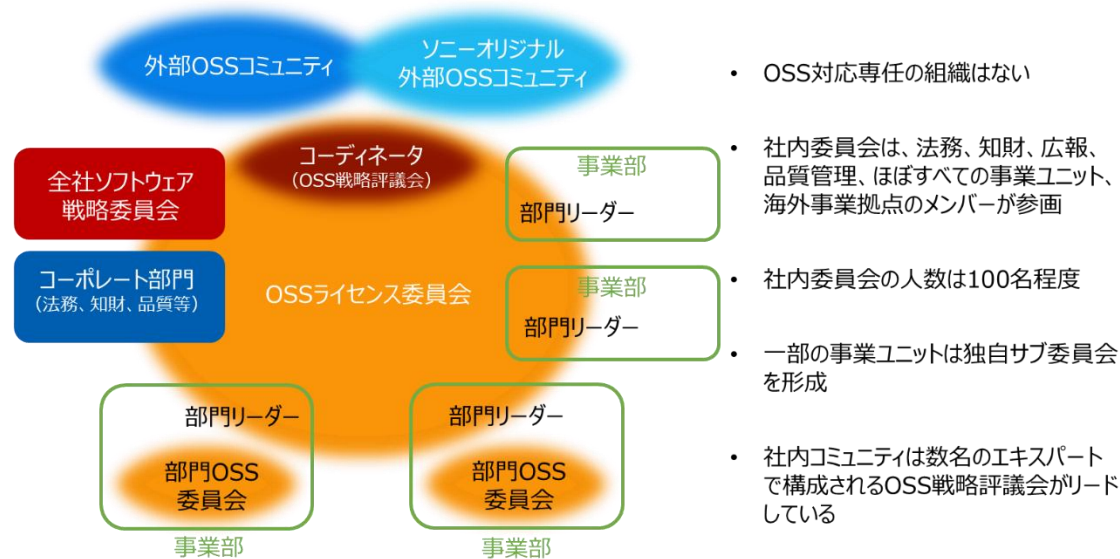


図 4.2-1 ライセンス対応における組織横断型体制のイメージ³⁸

なお、現在のライセンス対応における具体的な取組としては、OpenChain Japan WG のプラチナメンバーとして、その活動をリードすることによりサプライチェーン管理を強化している点があげられる。これは、ソニー製品の SoC³⁹ベンダ、OEM⁴⁰/ODM⁴¹ベンダ等のサプライヤから提供されるソフトウェアに含まれる OSS を、適切に把握しなくてはならないという課題意識が基となっている。ソニーでは、これまで特に海外サプライヤにおける OSS 使用状況の把握に苦労をしてきた経緯があった。その中で、ソニー単体でサプライヤに対応を働きかけるよりも、OpenChain Japan WG として、連携して働きかけることが有効であるとの考えに至っている。

例えば、OpenChain Japan WG で作成した成果物（OSS ライセンスコンプライアンスに関する啓発資料等）を、日本国内のみならず、英語、さらに中国語（繁体字、簡体字）、ベトナム語に翻訳の上で広くグローバルに発信をしている（図 4.2-2）。また、OpenChain Japan WG として、海外のメ

³⁸ OpenChain Japan WG 会合資料を基に作成

https://wiki.linuxfoundation.org/_media/openchain/openchainjwg_organization_lt_20180419_jpen.pdf

³⁹ System on a Chip: 一個の半導体チップ上にシステムの動作に必要な機能の多く、あるいは全てを実装するという設計手法

⁴⁰ Original Equipment Manufacturing (Manufacturer) : 他社ブランドの製品を製造すること、あるいはその企業

⁴¹ Original Design Manufacturing (Manufacturer) : 他社ブランドの製品を設計・製造すること、あるいはその企業

ーカー・サプライヤ等と積極的に交流する活動も続けている。そのかいもあり、韓国、中国、インド、ドイツ、英国において同様のWGが立ち上がる等、OSSのコンプライアンスに対する認識の高まりが広がっている。その中で、使用ソフトウェアの開示等の対応に海外サプライヤが理解を示す等の変化が現れ始めている。

現在、OpenChain Japan WGの活動は大企業中心となっているものの、今後はその活動の裾野を広げ、中小企業も取り込むことで、更に効果を高めることが期待されている。

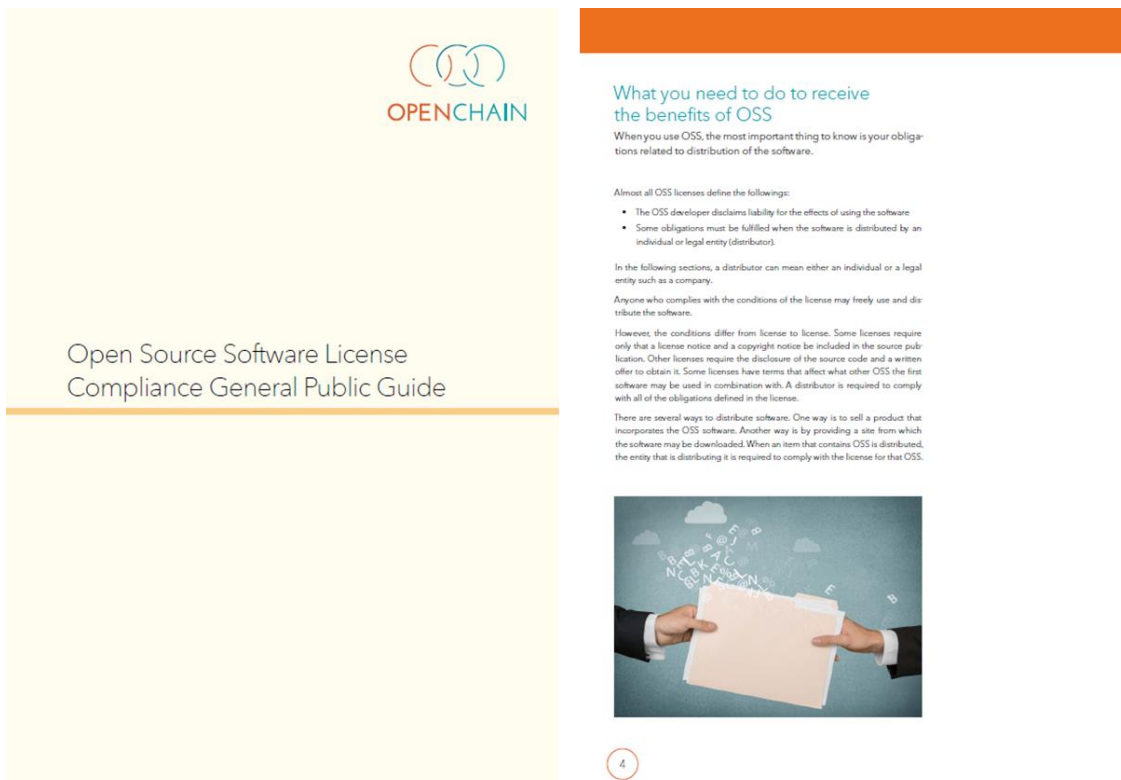


図 4.2-2 OSS ライセンスコンプライアンスの重要性を周知するリーフレット（英語版）⁴²

⁴² サイバー・フィジカル・セキュリティ確保に向けたソフトウェア管理手法等検討タスクフォース 第3回資料より抜粋

https://www.meti.go.jp/shingikai/mono_info_service/sangyo_cyber/wg_seido/wg_bunyaodan/software/pdf/003_04_00.pdf

オープンソースソフトウェア ライセンス遵守に関する一般公衆ガイド

<https://github.com/OpenChain-Project/Curriculum/tree/master/supplier-leaflet>

4.3. オリンパス ～ヒヤリ・ハット事象を契機とした全社的取組～

■ 企業情報・本事例集における位置づけ

本社	東京都新宿区
業種	精密機器
従業員数（連結）	35,174 人（2020 年 3 月末現在）
商流	製品商流/最終製品メーカー
観点	「ライセンス」「サプライチェーン管理」「個の能力・教育」「組織・体制」「コミュニティ活動」

■ 本事例のエッセンス

- 過去のヒヤリ・ハット事象を契機に、OSS ライセンスのコンプライアンス対応に向けたソフトウェア管理を全社的な取組へと発展させた。
- 「体制」「プロセス」「ツール」「教育」「サプライチェーン管理」の観点において、重点的に仕組み作りを行い、OSS ライセンスのコンプライアンス体制を整えた。

■ 背景・課題

オリンパスの製品は、医療、映像、科学と多岐にわたる。また、それら製品は多様なサプライチェーンからの部品・ソフトウェア供給を基に製造されている。その中で、納入品における OSS ライセンスのヒヤリ・ハット事象発生を契機に、全社統一で適切なソフトウェア管理の必要性が認識されていた。

■ 取組

<【ライセンス(L3)】【サプライチェーン管理(S3)】【個の能力・教育(P3)】【組織・体制(O3)】【コミュニティ活動(C3)】ヒヤリ・ハット事象を契機とした全社的取組>

オリンパスでは、過去、サプライヤーに開発依頼をしたソフトウェアにライセンス違反となりかねない形で OSS が用いられていたことが発覚するヒヤリ・ハット事象が発生していた。当該ソフトウェアを搭載した製品の出荷直前で至急 OSS ライセンスに係る対応を実施し、事無きを得たものの、この事象を受け、OSS ライセンスのコンプライアンスへの取組を徹底することとなった。

取組の方針として、OSS を使用するのであればライセンスを遵守すること、及び使用する意図のない OSS の混入を防ぐこと、の 2 点を念頭に置いた仕組み作りが目指された。具体的には、主に以下 5 つの観点において取組が進められた。

- ① 体制
- ② プロセス
- ③ ツール
- ④ 教育
- ⑤ サプライチェーン管理

以下、各取組の詳細について記載する。

① 体制

体制面においては、OSS ライセンスに係る全社委員会（OSS 推進会議⁴³）を立ち上げた。各事業部及びコーポレート部門から代表メンバーを出し、OSS コンプライアンスに関する施策の意思決定や情報共有等を行っている（図 4.3-1）。

特に、各事業部（医療、映像、科学）では、それぞれの市場の特性があるため、全社統一的な方針等の決定においては全社委員会における意見の擦り合わせが重要になってくる。また、全社委員会においては、原則的な事項（OSS 利用方針等）は意思決定するものの、具体的な部分は各事業部の裁量に委ねるといった形で柔軟性を持たせる工夫も行っている。

⁴³ メンバーの規模は 10 名程度で、月 1 回程度定期的に行っている。事務局は OSS 対応専任組織である OSS コンプライアンス室が担っている。

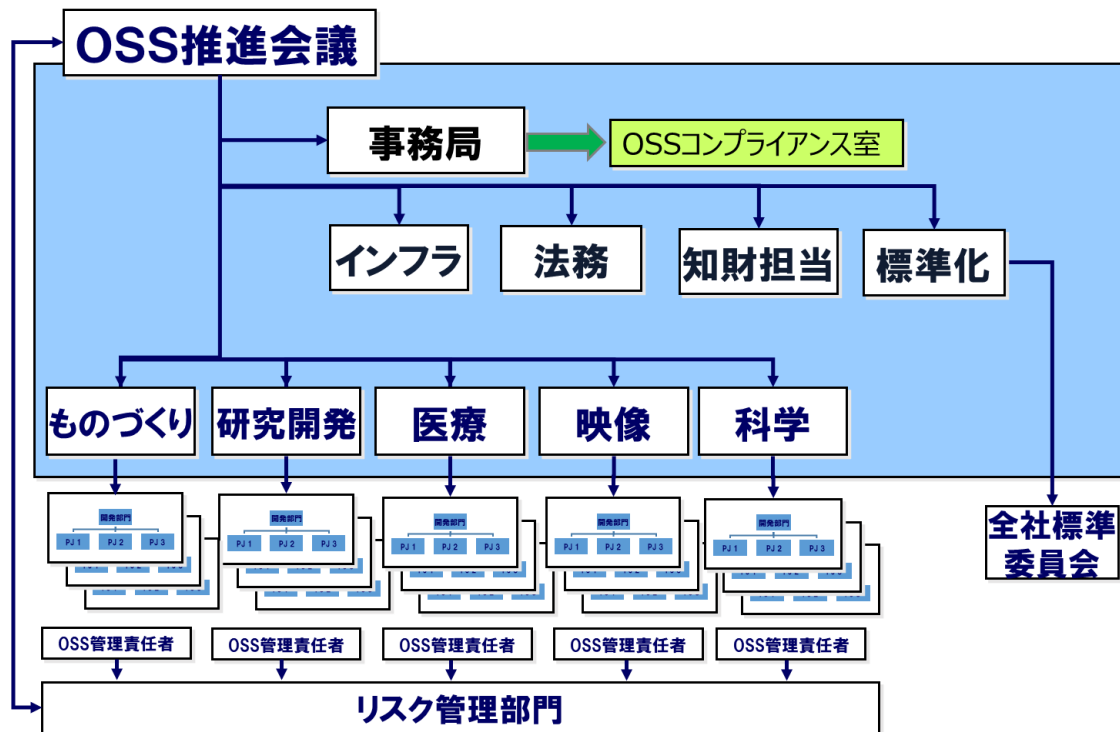


図 4.3-1 全社委員会の体制図⁴⁴

② プロセス

プロセス面においては、OSS 対応手順としての一連のプロセスを定め、既にあった全社の製品化に係るプロセス（以下、「標準プロセス」という。）に組み込んだ（図 4.3-2）。

⁴⁴ オリンパス提供資料を基に作成

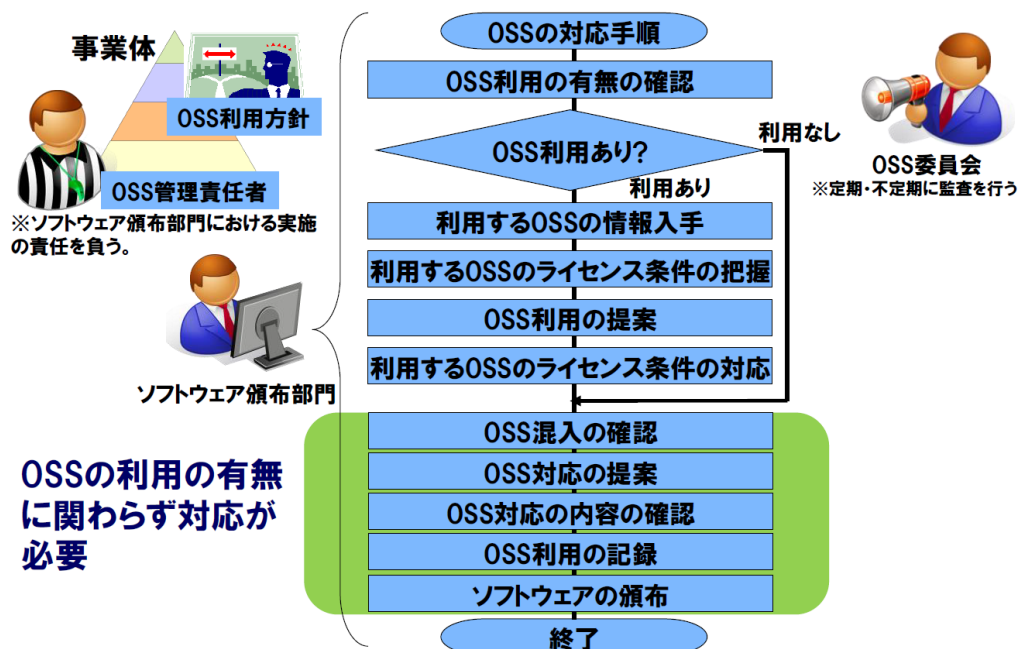


図 4.3-2 全社標準プロセスに含まれた OSS 関連プロセス⁴⁵

そのプロセスにおいては、OSS 利用申告の有無に関わらず、商用ツールを用いて、全ての製品に対して OSS の混入がないかを確認しなくてはならない。また、その確認結果（使用 OSS 情報）やライセンス対応方法は、定められたフォーマット（OSS 報告書（図 4.3-3））に基づき事業部で取りまとめ、OSS コンプライアンス室に提出をしなくてはならない。OSS コンプライアンス室では、それらの情報を最終確認し、記録、管理する。事業部によっては、それらの記録をソフトウェア脆弱性対応時に参照先として活用している。

新しいプロセスが追加されることに開発現場等から負荷が増えるとして反発はあったものの、もともと標準プロセスを遵守する企業文化は醸成されていたため、OSS に係るプロセスを標準プロセスに組み込むことで自然と遵守されていった。また、あるグループ会社で先行して本プロセスを実践し、その効果を明確にすることで納得感を得る工夫も行った。

⁴⁵ オリンパス提供資料より抜粋

#	OSS ごとの情報				
1	基本情報				
	名称				
	バージョン				
	開発者（著作権者）名				
	取得元の URL				
	ライセンス条件				
	ライセンス名称				
	ライセンスタイプ	GPL タイプ	LGPL タイプ	MPL タイプ	BSD タイプ
	利用方法				
	リンク形式	静的リンク	動的リンク	プロセス間 通信	独立した プログラム
	ソースコードの改変	あり		なし	
	ライセンス条件の具体的対応方法				
	著作権の表示方法				
	ライセンス本文の表示 方法				
	免責事項・無保証・特 許などに関する記載の 表示方法				

図 4.3-3 OSS 報告書に記載する OSS に関する情報⁴⁶

③ ツール

ツールについては、2 つのツールを活用している。1 つは、前述のとおり、OSS 混入を確認する商用ツールである。なお、②と同様、あるグループ会社で先行してそのツールを活用し、効果を確認した上で全社展開するといった工夫も行った。現在では、グループ全体でそのツールを利用しなくてはならない。

もう 1 つは、OSS に関する情報集約のためのナレッジサイトを構築している。OSS 利用に際しての注意点、ライセンスの種類、社内の製品の OSS の利用状況等の情報を提供している。

④ 教育

OSS コンプライアンス遵守に係るプロセスや各種ツールの使い方を周知するための教育プログラムも充実させている（図 4.3-4）。実務者のみならず、経営層⁴⁷やパートナー向けの OSS 教育⁴⁸も実施している。また、英語の教材等も用意することで、海外拠点の社員も教育を受けることが可能である。

⁴⁶ オリンパス提供資料より抜粋

⁴⁷ 執行役員及び事業部の幹部向けに技術者認定団体のトップによる講演会等を開催

⁴⁸ 社員向けの基礎教育資料「OSS 基礎」（図 4.3-4 内）と同様の内容を一部社外向けにカスタマイズした教育メニューである。

教育体系		講演会	集合研修	e-Learning	資料配布
経営者向けOSS研修	戦略立案や方針策定、OSS利用を判断する経営者向け				
OSS検出ツール研修	OSS検出ツールを使用する担当者向け				
SW開発者向けOSS研修	OSS利用を提案、OSS利用状況を把握する担当者向け				
法務担当者向けOSS研修	OSSを利用する案件を担当する担当者向け				
知財担当者向けOSS研修	OSSを利用する案件を担当する担当者向け				
OSS基礎	OSSを利用したソフトウェアを開発、頒布する担当者向け				
SW開発委託者向けOSS基礎	OSSを利用した販促品の開発委託やOEMの仕入れを担当する担当者向け				
ユーザー対応者向けOSS基礎	製品購入したお客様と接する営業やサポート担当者向け				
パートナー向けOSS基礎	OSSを利用したソフトウェアを開発するパートナー向け				

図 4.3-4 多様な対象者を想定した教育メニュー⁴⁹

⑤ サプライチェーン管理

サプライチェーン全体に対して OSS コンプライアンスの啓発を実施する目的で、OpenChain Japan WG⁵⁰に参加している。これは、前述のようなヒヤリ・ハット事象を防ぐには、社内、グループ内の取組だけではなく、サプライチェーンへの働きかけが必要との課題意識が契機となっている。

オリンパスでは、サプライヤからの部品・ソフトウェア納入の際に、確認書（OSS 利用の有無の確認書（図 4.3-5））の提出を求めている。このような対応等に対するサプライヤの理解を得るため、OpenChain Japan WG を活用して発信を行っている。具体的な発信の一例として、OpenChain Japan WG において、前述の OSS 報告書フォーマットの運用について共有を行ったことがあげられる。それがきっかけとなり、OpenChain Japan WG 発信の SBOM フォーマット（SPDX Lite）検討につながった。

⁴⁹ オリンパス提供資料より抜粋

⁵⁰ OpenChain Japan WG の具体的な取組については、トヨタ自動車の事例に記載

_____ 様 オープンソースソフトウェアの利用の有無の確認書 時下、ますますご清祥のこととお慶び申し上げます。この度貴社からの納入物（製品、部品、ユニット、ソフトウェア、またはサービスなど）に、公然利用可能なソフトウェアであって、一般的に「オープンソースソフトウェア」と呼ばれているソフトウェア（以下 OSS と呼ぶ）が利用されているか否かを教えて頂きたくご連絡申し上げます。つきましては納入物における OSS の利用の有無について、本信末尾の回答欄にご記入いただき、貴社御記名捺印の上、弊社宛にご送付賜りたくお願い申し上げます。また、納入物に OSS が利用されている場合には、さらに以下の情報を納入物ごとに提供していただけますようお願い申し上げます。 記 1) OSS を含んでいる納入物の名称 2) 含まれている全ての OSS の名称のリスト 3) 含まれている全ての OSS のライセンス条文、または利用条件 4) 著作権に関する告知文章（必要な OSS に対して） 5) ソースコード、オブジェクトコード、およびそれらに付随した情報（必要な OSS に対して） 6) オリンパスが当該 OSS またはその派生物を複製、翻案、改変または頒布するにあたり必要なその他の情報 以上 上記質問事項につき、以下のとおり回答いたします。 納入物の名称 : _____ 納入物における OSS 利用の有無 : <u>（ 有 ・ 無 ）</u>

図 4.3-5 OSS 利用の有無の確認書⁵¹

⁵¹ オリンパス提供資料より抜粋

4.4. 日立製作所 ～製品化の過程における徹底した OSS 管理～

■ 企業情報・本事例集における位置づけ

本社	東京都千代田区
業種	電気機器
従業員数（連結）	約 301,000 人（2020 年 3 月末現在）
商流	製品商流/最終製品メーカー
観点	「ライセンス」「コミュニティ活動」

■ 本事例のエッセンス

- IT 製品においては、その製品化の過程で、ソフトウェア部品構成表（SBOM）による管理と商用ツールによる OSS のチェックを必ず実施することで、OSS 管理の一元化・効率化、及びライセンスコンプライアンスの徹底を実現している。
- コミュニティ活動における幅広い貢献を行い、その成果を自社の事業展開に還元している。

■ 背景・課題

日立製作所は、1990 年代後半より、Linux 等へのコミュニティ貢献をはじめ、OSS の普及促進に尽力してきた。現在でも、OSS の利活用を成長戦略の重要な要素と位置づけ、OSS に対する取組を一層強化している。

■ 取組

＜【ライセンス^(L4)】製品化の過程におけるソフトウェア部品構成表（SBOM）活用と商用ツール利用＞

日立製作所では、OSS 対応専任組織である OSS ソリューションセンタとコーポレート部門（知的財産、法務、調達、QA⁵²部門）等が連携し、OSS 利活用の普及促進及び支援等を行っている。また、OSS セキュリティにおける技術的な側面（脆弱性対応、ツール開発・運用等）については、セキュリティ事業統括本部と連携して対応している。

⁵² Quality Assurance : 品質保証

日立製作所における OSS の利活用は多岐に渡る。特に、情報処理機器や通信機器、ソフトウェア等の IT 製品、及びサービス分野、SI 分野においては、多くの OSS が使用されているため、ライセンスコンプライアンスの面等から、OSS 管理を徹底している。具体的には、以下の 2 点の取組を行っている（図 4.4-1）。

- ① ソフトウェア部品構成表（SBOM）を活用したシステムによる管理
- ② 商用ツール活用による OSS の効果的な把握

以下、各取組の詳細について記載する。

① ソフトウェア部品構成表（SBOM）を活用したシステムによる管理

日立製作所は、自社にとって最適な管理を実現するため、SBOM を活用した管理システムを自社開発し、2013 年にソフトウェア開発部門で稼働させた。現在では、IT セクター（IT 製品、サービス分野、SI 分野）で活用されている。

現在の IT セクターにおいては、この管理システムによって OSS が一元管理され、OSS 利用に関する社内外の情報を集約することで、効率的なライセンス調査等が実施できている。また、使用 OSS のデータベース化やワークフローシステムの機能により、トレーサビリティの確保や手続き面の効率化も可能となっている。

開発チームは、製品開発段階において、使用する OSS を管理システムに登録する。登録された OSS は、過去の使用実績等も踏まえながら、審査ワークフローシステムを通じて、ライセンス面で使用することが適切であるかチェックされる。各プロジェクトの使用 OSS は、SBOM が作成された上でデータベースに登録、使用実績として蓄積され、次期開発にフィードバックされる。

② 商用ツールの活用による OSS の効果的な把握

日立製作所の IT セクターにおいては、開発初期段階における OSS のライセンス調査や、開発委託品⁵³及びリリース前の製品に未認識の OSS が含まれていないかチェックする目的で商用ツールを導入している。これにより、管理システムへの登録だけでは把握しきれなかった OSS によるライセンス違反を防止している。また、商用ツールによって検知された OSS は、前述の管理システムに自動登録される。なお、製品リリース後のソフトウェア脆弱性対応においても商用ツールの機能（脆弱性発生時のメール通知機能）が活用されている。

⁵³ 開発委託品については、OSS を使用する場合は事前通知する旨を委託先との基本契約書に盛り込むことや、開発委託をする際に OSS の使用可否を発注の条件書に記載する等の対応も行っている。

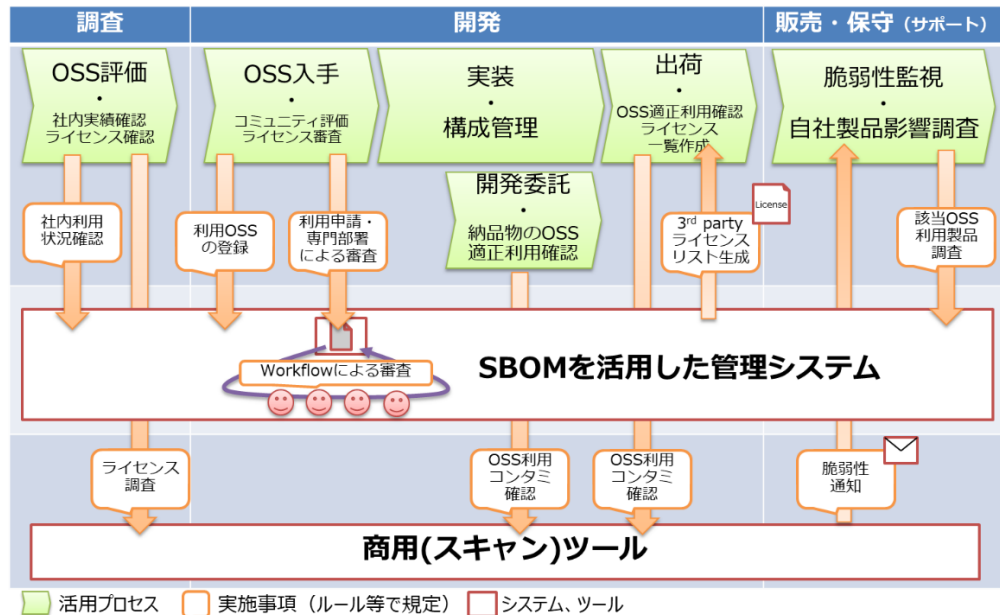


図 4.4-1 開発プロセスにおける管理システム及び商用ツールの活用状況⁵⁴

IT 製品においては、これらの管理システム及びツールを用いることが社内規則として定められており、製品化に至るための必須実施項目となっている。一方で、開発現場からは OSS の登録作業等に作業負荷がかかるといった声もあがっていたため、ソフトウェアの開発支援を主に行っている部署が一部 OSS の登録を代行するメニューを設ける等の工夫も行っている。

さらに、OSS 管理の中で、システム、ツール、情報サイト等における OSS の名称が統一されておらず、その判別に苦勞する場面が発生している。その対応として、AI を用いて各名称を整理する仕組みの構築についても検討を進めている。

現在、OT（Operational Technology）分野の事業でも、独自のソフトウェア構成管理や脆弱性対応の仕組みはあるものの、IT 製品と同等のレベルにまで達するよう、これらの管理システム及びツール利用を拡大している最中である。

<【コミュニティ活動(c4)】コミュニティ活動における幅広い貢献>

⁵⁴ 日立製作所提供資料を基に作成

日立製作所は、OSS の利活用の更なる拡大を目的に、OSS の開発、調査・検証、普及促進に係る各種コミュニティ活動に、社員をメンバーとして参画させる等、幅広い貢献をしている（図 4.4-2）。

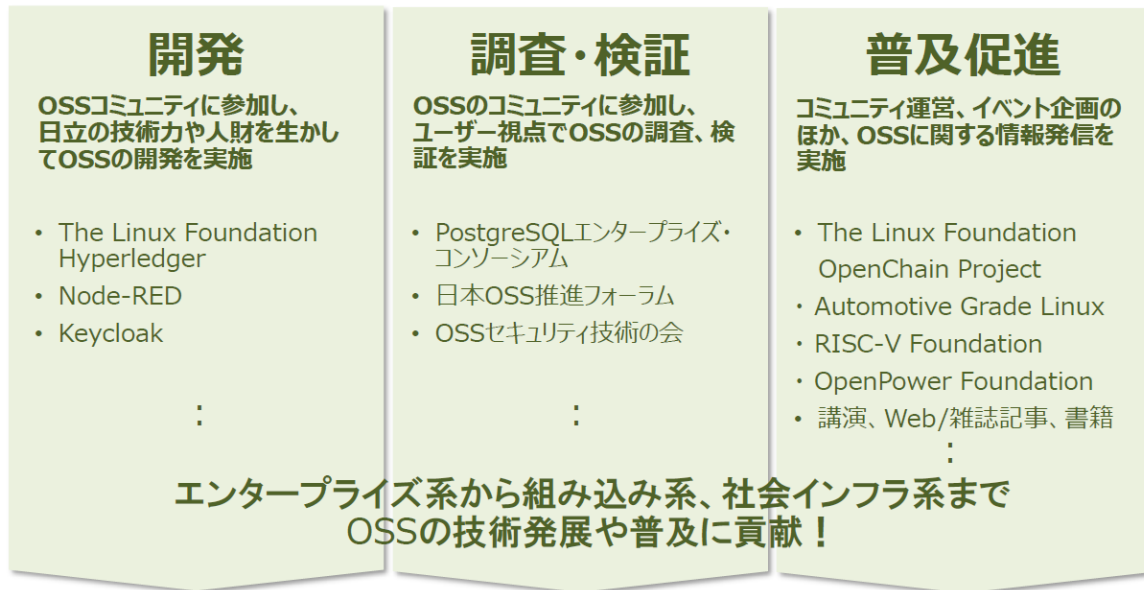


図 4.4-2 日立製作所が参画する主なコミュニティ⁵⁵

日立製作所では、自社の持つ多様な事業領域との関連性を見極めながら、多数のコミュニティに参画している。結果的に、それらのコミュニティ活動の成果は、日立製作所自身の事業展開にも還元されている。具体的には、それらのコミュニティにおける OSS 開発の実績を社外講演等でアピールすることで顧客から案件の引き合いを得たり、コミュニティ活動の中で把握した先進的な取組やベストプラクティスを自社プロセスの効率化や研修内容の見直し等に繋げている。

また、国内外の様々なイベントでの講演や各種メディアを使った情報発信⁵⁶を積極的に実施する等、日立製作所は OSS 利活用の普及促進の流れをリードしている。OSS の利活用支援のソリューションも提供しており、その提供先は金融、公共、通信、ヘルスケア等と多岐に渡っている。

⁵⁵ 日立製作所提供資料より抜粋

⁵⁶ OSS に対する自社の取組や OSS 活用の具体的なユースケースの紹介等を行っている。

4.5. オムロン ～PSIRT の連携を通じた OSS 対応～

■ 企業情報・本事例集における位置づけ

本社	京都府京都市
業種	電気機器
従業員数（連結）	28,006 名（2020 年 3 月末時点）
商流	製品商流/最終製品メーカー
観点	「脆弱性対応」「サプライチェーン管理」「個の能力・教育」「組織体制」

■ 本事例のエッセンス

- 本社部門と各事業体からメンバーを集めて PSIRT を組織し、役割分担を明確にした上で連携をしながら、OSS に係る管理や教育、脆弱性対応等を実施している。
- 本社部門と各事業体の連携のための協議体としてワーキンググループを設置し、製品セキュリティ等に係る改善策やグループ統一のルールに係る議論、定期的な情報交換等を実施している。

■ 背景・課題

オムロンでは、制御機器・FA システム、電子部品、ヘルスケア、社会システム等、多様な事業ポートフォリオを、一部分社化も含めたカンパニー制にて展開している。その中で、OSS 対応等を含むセキュリティの取組について、本社部門と各事業体が適切に連携しつつ実施する必要がある。

■ 取組

<【脆弱性対応(V5-1)】【サプライチェーン管理(S5)】【個の能力・教育(P5)】【組織体制(O5-1)】PSIRT 体制及び本社部門と各事業体の連携>

オムロンでは、本社部門と各事業体（カンパニー及び事業子会社）からメンバーを集めて PSIRT が組織されており、OSS 対応等を含む製品セキュリティへの取組を進めている。本社部門では、製品の品質、セキュリティに対するグループ統一のルール策定、及び調整を実施している。一方、各事業体では、担当製品に係るセキュリティについて実務的な対応等を行う体制となっている。

本社部門と各事業体は、定期的に製品セキュリティに関するワーキンググループ（協議体）を開催し、

製品セキュリティ等に係る改善策やグループ統一のルールに係る議論、脆弱性情報等に対する対処事項の周知徹底、及びその他情報交換を実施している（図 4.5-1）。

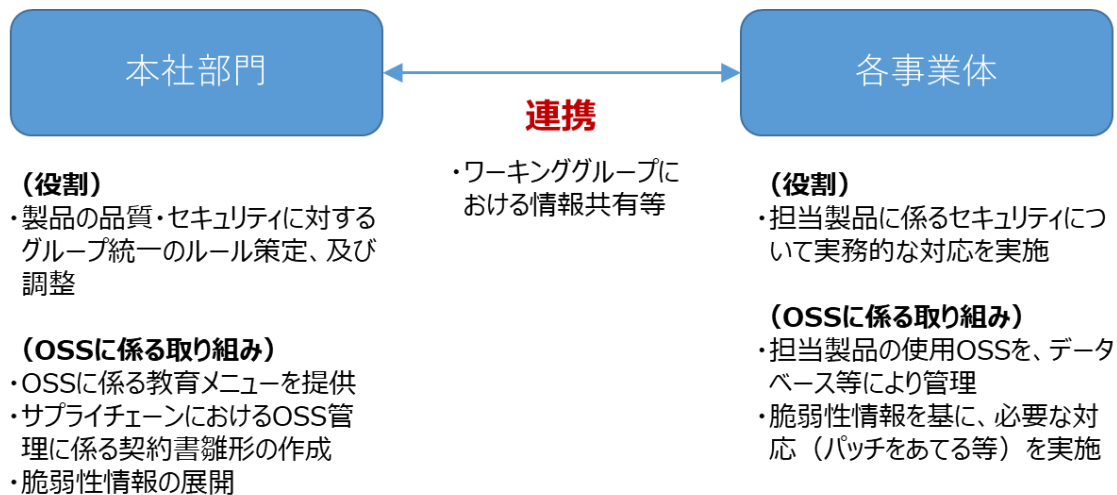


図 4.5-1 本社部門と各事業体の役割分担及び OSS に係る取組⁵⁷

具体的な OSS 対応に係る取組は、各事業体において、担当製品の使用 OSS をデータベース等によって管理することにより実施されている。事業体によってセキュリティの要求レベルや OSS の使用頻度が異なっているため、セキュリティに対するグループ統一のルールには沿いつつも、OSS の管理粒度やその手法については事業体ごとに判断している。商用ツールを用いて管理を実施する事業体もある。

一方、本社部門においては、各事業体の OSS 管理を補完している。具体的には、開発者向けに、OSS の概要、OSS の管理リスク（OSS が意図せず混入するケース等）、ライセンスコンプライアンス等を中心とした、OSS に係る教育を実施⁵⁸している。教育の内容に応じて、法務部門や開発プロセス改善部門の担当者が講師を担う等、より実務に即する工夫もなされている。また、マネージャー層等、開発者以外が受講するケースもあり、全社のリテラシーの底上げにも寄与している。

また、本社部門では、サプライチェーンの中で意図せず OSS が混入することのないよう、外部の委託開発先と取り交わす契約書の雛形も整備している。雛形では、委託開発先に対し、OSS を原則使用しないこと、使用する場合にはその OSS を通告すること等を定めている。各事業体の担当者は、この雛形を活用しつつサプライチェーンの OSS についても適切に管理している。

⁵⁷ ヒアリングを基に作成

⁵⁸ 対面、もしくは Web による教育を実施している。

<【脆弱性対応(V5-2)】【組織体制(O5-2)】ソフトウェア脆弱性対応に係る PSIRT の連携>

オムロンでは、本社部門の PSIRT 担当者が定常的に OSS の脆弱性情報を収集しており、OSS の脆弱性が判明した際には、関連する事業体の PSIRT 担当者にその情報をメールで展開している。情報セキュリティ早期警戒パートナーシップから取得した脆弱性情報にはテクノロジーキーワード⁵⁹が付されているため、そのキーワードを基に関連する事業体を特定する他、本社部門の PSIRT 担当者の判断の下、重要な脆弱性については対応の優先度を高めるようフラグを付けて展開する等の工夫を行っている。各事業体では、展開された情報を基に対象の OSS にパッチを当てる等の対応をとっている。また、その対応状況はリスクの程度に応じて本社部門にも報告がされる。なお、脆弱性情報は、情報セキュリティ早期警戒パートナーシップだけでなく、社内外からの情報が本社部門と各事業体で連携され、脆弱性リスクの評価及び対策が実施されるルールになっている。

現在、オムロンでは、各事業体で実施している脆弱性情報の管理を本社部門にて一元化するシステムの導入を検討している。その目的は、本社部門が一括して管理することによる業務効率化と、過去の OSS に係る対応履歴を全グループで参照可能にすることによる対応判断の精度向上にある。あわせて、脆弱性情報取得時の周知メールの展開についても、当該システムと連携させ、そのフローを自動化させたいと考えている。

⁵⁹ JPCERT/CC が、脆弱性情報を受け付ける各ベンダに対して登録を求めている技術用語。ベンダごとにリスト化されており、脆弱性関連情報が報告された際、JPCERT/CC が連絡すべきベンダを選出するための参考情報となる。

4.6. 東芝 ～グループにおける一貫した OSS 対応体制～

■ 企業情報・本事例集における位置づけ

本社	東京都港区
業種	電気機器
従業員数（連結）	125,648 名（2020 年 3 月末時点）
商流	製品商流/最終製品メーカー
観点	「脆弱性対応」「組織体制」「選定評価」「ライセンス」「保守・品質保証」「コミュニティ活動」

■ 本事例のエッセンス

- ・ 東芝 SIRT が一次窓口となり、グループにおけるソフトウェア脆弱性対応に関する一貫した対応体制を構築している。
- ・ 共創ソフトウェア開発プラットフォームを構築し、開発成果や OSS コンプライアンスプロセスの成果の再利用による効率化、品質向上、OSS 利活用促進等を図っている。
- ・ 自社の強みである社会インフラ領域で OSS の利活用を拡大させるため、関連するコミュニティ活動に参加している。

■ 背景・課題

東芝は、エネルギー、社会インフラ、電子デバイス、デジタルソリューションを中心とした事業を展開しており、特に社会インフラ領域におけるデジタルトランスフォーメーションを推進している。クリティカルな品質が求められる事業領域の中で、多くの人が安心して東芝の製品・システム・サービスを利用できるよう製品セキュリティに対する取組を強化している。

■ 取組

<【脆弱性対応(v6)】【組織体制(o6)】東芝 SIRT を中心としたソフトウェア脆弱性対応と SIRT 支援システム>

東芝では、CISO⁶⁰が製品セキュリティに関する権限を持っている。製品セキュリティの強化に向けて、CISOの下、サイバーセキュリティセンターに設置した東芝 SIRT⁶¹及び主要なグループ会社の PSIRT が連携するサイバーセキュリティ管理体制を構築し、ソフトウェア脆弱性やセキュリティインシデントへの対応、セキュア開発マネジメント等に重点を置いて取り組んでいる。特に、ソフトウェア脆弱性対応においては、情報セキュリティ早期警戒パートナーシップに参加し、外部機関と積極的に連携しながら、迅速かつ確実な対応をするために東芝グループ全体で一貫した対応体制を構築している。

具体的には、東芝グループの製品・システム・サービスに何らかの脆弱性が発見された場合、東芝 SIRT が一次窓口として脆弱性情報を集約し、社内外への情報共有を図る体制となっている。製品に使用されている OSS 等、ソフトウェアの脆弱性が判明した場合も、東芝 SIRT は、その製品の事業主体であるグループ会社の PSIRT、及び対象製品の事業部と連携し、その対応にあたる（図 4.6-1）。対応した結果を、自社 Web サイトや JVN 等において公表する取組も実施している。

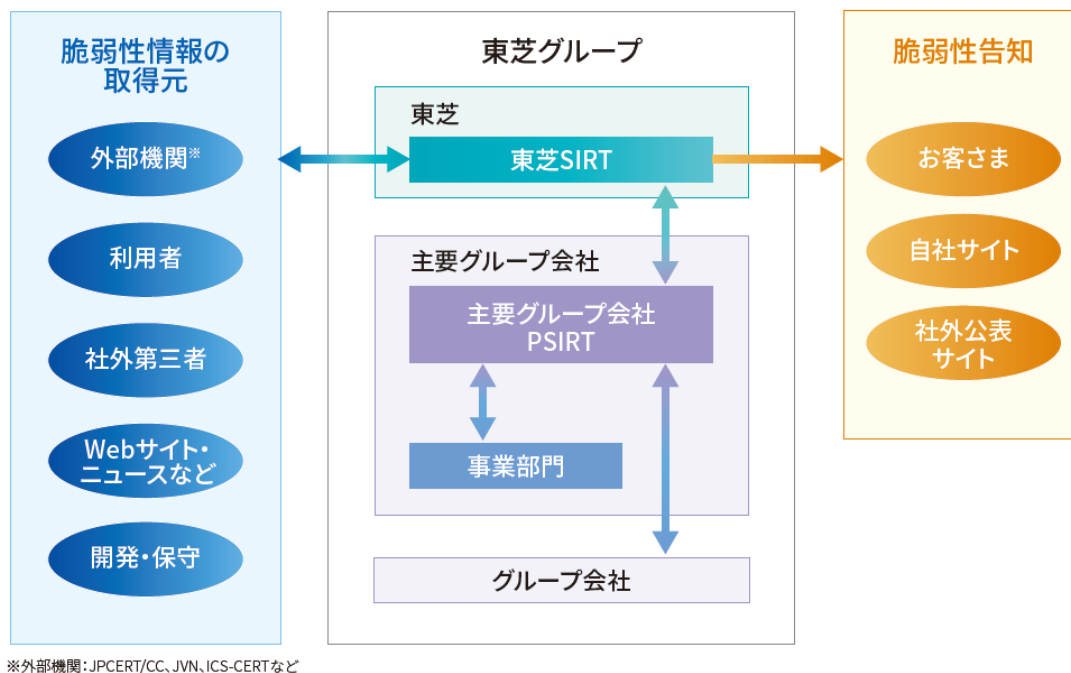


図 4.6-1 東芝グループ脆弱性対応の体制⁶²

⁶⁰ Chief Information Security Officer：最高情報セキュリティ責任者

⁶¹ Security Incident Response Team：コンピューターシステムなどにセキュリティ上の脅威が発生した際に対応する組織

⁶² 東芝グループ サイバーセキュリティ報告書 2020より抜粋

<https://www.global.toshiba/content/dam/toshiba/migration/corp/securityPrivacyAsse>

また、東芝では、東芝 SIRT が収集したソフトウェア脆弱性情報をグループ会社や事業部に通知するシステム（SIRT 支援システム）を構築している。これは、登録されたソフトウェア構成情報と脆弱性情報を、CPE⁶³もしくは製品名称によって紐づけ、関係個所へ自動的に通知するシステムである（図 4.6-2）。ソフトウェア構成情報については、各グループ会社、各事業部が管理しており、SIRT 支援システムにあらかじめ登録しておく。グループ会社が自社でソフトウェア構成情報を管理するシステムを持っている場合は、その情報を自動連携し SIRT 支援システムに登録できるようにする取組も進めている。脆弱性情報は、JVN や NVD⁶⁴等の脆弱性情報データベースと API 連携し自動収集するほか、顧客等から報告されたものについては手入力により登録する。

また、通知されたソフトウェア脆弱性情報が、グループ会社や事業部でどのように対応されたかをシステム上で管理できる機能を備えており、設計・開発部門、事業部、主要グループ会社のそれぞれの階層に責任者を設け、その責任者を中心にアフターフォローする仕組みとしている。

2020 年 10 月現在、SIRT 支援システムは試行運用中であり、各グループ会社、各事業部にて活用が始まったところである。特に、セキュリティ感度の高い製品を扱うグループ会社（社会インフラ系等）においては、ツール等も用いながらソフトウェア構成情報を漏れなく取得し、SIRT 支援システムに登録を行うなど、活用が進んでいる。

ts/security/files/CyberSecurityReport2020.pdf

⁶³ Common Platform Enumeration : ベンダ名、製品名、バージョンまでを一意に特定できる ID

⁶⁴ National Vulnerability Database: アメリカ国立標準技術研究所（NIST）が管理する脆弱性の情報を集めたデータベース

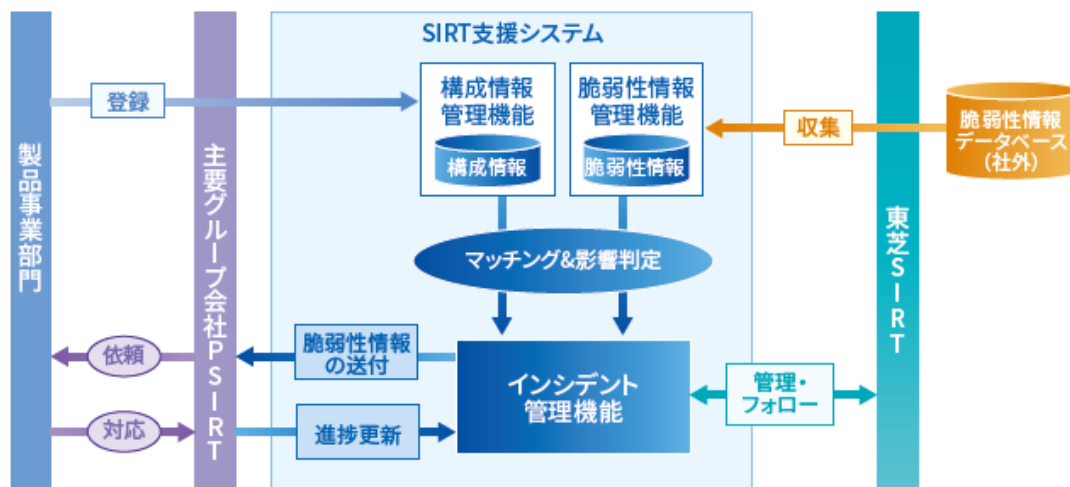


図 4.6-2 SIRT 支援システムの概要⁶⁵

<【選定評価(S6)】【ライセンス(L6)】【保守・品質保証(M6)】共創ソフトウェア開発プラットフォームを活用した開発成果及び OSS の再利用>

東芝では、更なる OSS 利活用を見据え、ソフトウェア技術センターが中心となり、共創ソフトウェア開発プラットフォームを構築している。これは、先行している開発で得られた技術的成果や OSS コンプライアンスプロセスの成果を記録・管理し、別の開発で再利用することを可能とする仕組みである。これにより、製品・システム・サービスの開発工数削減や機能改善、品質向上、及び OSS 利活用の促進を図ることが可能となる。共創ソフトウェア開発プラットフォームは、①ソフトウェア開発管理システム、②ソフトウェア資産管理システム、の 2 つから構成されており、従来のソフトウェア開発運用プロセスと、OSS を利用するための OSS コンプライアンスプロセスの手順を同時に遂行させることが可能となっている(図 4.6-3)。共創ソフトウェア開発プラットフォームは、東芝グループ全体での利用を見据えて整備が進められている。

⁶⁵ 東芝グループ サイバーセキュリティ報告書 2020 より抜粋

<https://www.global.toshiba/content/dam/toshiba/migration/corp/securityPrivacyAssets/security/files/CyberSecurityReport2020.pdf>

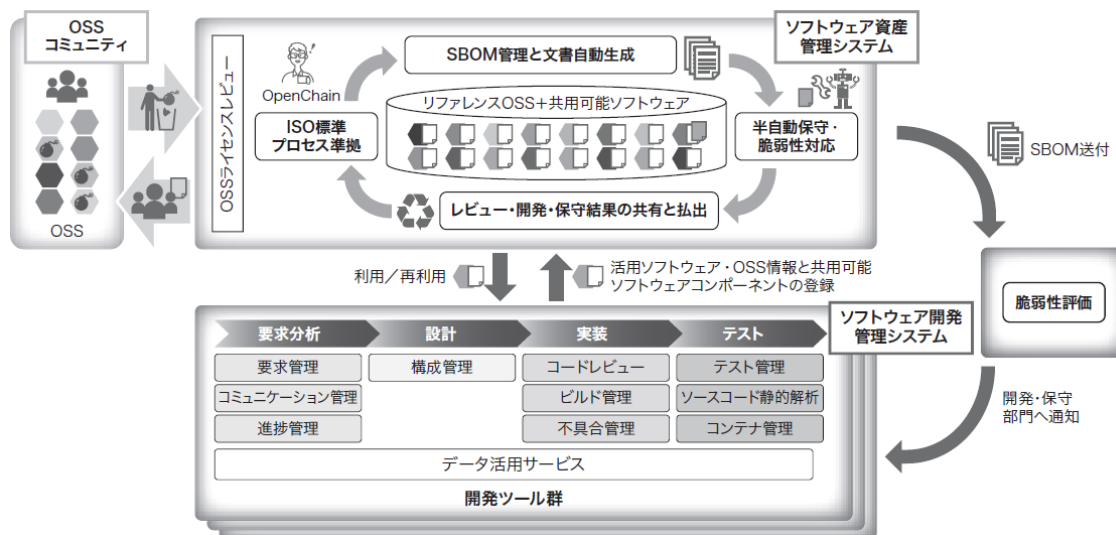


図 4.6-3 共創ソフトウェア開発プラットフォームの構成⁶⁶

共創ソフトウェア開発プラットフォームを構成する 2 つのシステムの機能は以下のとおりである。

① ソフトウェア開発管理システム

ソフトウェア開発管理システムは、OSS を活用したソフトウェア開発ツールチェーンで構成されており、ソースコード静的解析や、トレーサビリティ確保を効率化する要求管理、脆弱性検出等の開発管理機能を持つツールがサービスとして提供されている。

② ソフトウェア資産管理システム

ソフトウェア資産管理システムは、OSS を利用する際のライセンスレビューから、ライセンスコンプライアンスにおいて必要となる SBOM によるソフトウェア管理等、OSS に係る一連のプロセスで使用するツール等が提供されている。

現在、東芝では、特に SBOM によるソフトウェア管理に力を入れており、各グループ会社に対してソフトウェア構成管理の重要性を啓発することをはじめ、セキュリティ対応の際にも活用できる高い精度の SBOM 運用を目指して標準化や運用の改善を図っている。また、各事業部で実施される OSS 等のライ

⁶⁶ 東芝レビューVol. 75 No. 5（2020 年 9 月）「開発効率を向上させる共創ソフトウェア開発プラットフォーム」より抜粋

https://www.global.toshiba/content/dam/toshiba/migration/corp/techReviewAssets/tech/review/2020/05/75_05pdf/a07.pdf

センスレビューの際には、知財部門等の関係者も出席し、「コミュニティの活動状況」「採用実績はあるか」といった観点や「評価で使用するか」「製品で使用するか」といった用途の観点を踏まえて OSS の採用可否を丁寧に判断している。なお、この際に必要に応じてソフトウェア技術センターが支援する。

<【サプライチェーン管理(s6)】開発委託と受入>

ソフトウェアによらず外部から調達するものについては、利用許諾を含む知的財産関係、脆弱性や品質などを確認することとしている。そのため、OSS を含む第三者によるソフトウェアを利用、調達する際も、その構成などを含めて確認している。こうした点は、自社が利用する場合でも、顧客等へ提供する場合でも、変わることはない。従って、開発初期や企画段階から、OSS のライセンスや脆弱性については重要事項として扱うこととなる。OSS の選定や利用においては、OSS のライセンス条件や機能、および事業運用に関する要件などの諸事情を考慮することが必要である。さらに、脆弱性の対応や将来的なアップデートへの対応について顧客や取引先との合意を形成することが重要である。

<【個の能力・教育(p6)】OSS 利活用に関する教育>

東芝では、OSS に関するマニフェストを定め、OSS コミュニティの知的財産を受け利活用する以上、OSS コミュニティに還元し、オープンソースのエコシステムの発展に寄与すべきであるといったカルチャーや考え方等の浸透を図っている。そうした理念に基づき、すべての職種向けの入門的な教育をはじめ、主にソフトウェア開発者向けのより詳細な内容を含む基礎的な教育など、実務に応じた段階的な教育を用意している。また、各グループ会社は必要に応じて独自の教育を行っている。例えば、入門的な教育では、オープンソースの特徴や企業活動における OSS の重要性とあわせて、知的財産、ライセンスコンプライアンスや脆弱性管理など、活用での留意事項に触れている。また、基礎的な教育以降では、オープンソースコンプライアンスのためのプロセスマネジメント標準である ISO/IEC 5230 (OpenChain Specification) を考慮した OSS を活用するためのプロセスマネジメントとあわせて、GPL 等の具体的な事例を示してオープンソースライセンスへの理解を深めるようにしつつ、ライセンスコンプライアンスの実務を踏まえた内容を盛り込んでいる。さらに、東芝では、オープンソースの開発スタイルを組織に導入するインナーソースの推進にも取り組んでおり、OSS を含むソフトウェア開発のトレーニングプログラムとして、ハンズオン形式の実践的な教育も用意している。

<【コミュニティ活動(c6)】OSS の利活用に向けた社内での連携及びコミュニティを通じた社外とのコラボレーション>

東芝では、OSS 利活用に係る体制として、グループ内の様々な事業部（法務等のコーポレート部門

も含む) からなる委員会を設置している。OSS ライセンスコンプライアンスの事例等の情報共有や、ポリシーやガイドラインの整備等、幅広く議論を行っている。

また、コミュニティへの参加等、社外とのコラボレーションも行っている。自社で利用している OSS ツールの開発コミュニティに参加し、機能改善等の提案を実施したり、OpenChain⁶⁷に参加し、サプライチェーンにおける OSS コンプライアンスの実現に寄与している。

さらに、東芝が中心メンバーの 1 つとなり、Linux Foundation 配下に Civil Infrastructure Platform (CIP) という OSS プロジェクトを立ち上げている。これは、社会インフラのような長期的に使われ、高い信頼性の求められるシステムにおいて OSS を安全に使っていくために共通の課題を解決していくとするプロジェクトである。一例として、社会インフラシステムや工場などで使われている産業機器では、OS として Linux を利用するケースが増加しているものの、Linux のサポート期間が数年程度である事が課題となっている。CIP では、10 年以上の長期間運用を想定して OSS を使用できる仕組み等を議論している。東芝は、社会インフラ領域における事業を強化する中で、CIP の活動を通じて、OSS 利活用メリットの最大化を目指している。

⁶⁷ OpenChain の活動概要については、トヨタ自動車の事例に記載

4.7. デンソー ～サプライチェーン全体における最適な OSS 管理～

■ 企業情報・本事例集における位置づけ

本社	愛知県刈谷市
業種	輸送用機器
従業員数（連結）	170,932 人（2020 年 3 月末現在）
商流	製品商流/サプライヤ
観点	「ライセンス」「サプライチェーン管理」「脆弱性対応」

■ 本事例のエッセンス

- ・ サプライチェーン全体のライセンス管理のため、自社内のルール整備、Tier2⁶⁸サプライヤとの取決めとその調整・合意、サプライヤと共有するためソフトウェアリストのフォーマットの統一を行っている。
- ・ OSS の脆弱性対応について、商用ツールや外部リソースの活用をしつつ脆弱性調査システムを運用している。

■ 背景・課題

自動車業界は、最終製品メーカーのみならず、重層的にサプライヤが存在し、多数の企業がサプライチェーンを形成している。主に Tier1 のサプライヤであるデンソーにおいては、自社における製品開発で OSS を使用することあれば、Tier2 以降のサプライヤから供給される製品に OSS が使用されている場合もあり、それらの使用状況を統合的に管理する必要がある。

■ 取組

＜【ライセンス(L7)】【サプライチェーン管理(S7)】サプライチェーン全体を含む OSS ライセンスに係る取組＞

デンソーでは、カーナビゲーション等のマルチメディア製品において、先行して OSS の利活用が始まり、徐々にその利活用製品の範囲を拡大させてきた。その中で、OSS ライセンスについて適切な対応を行うために、全社的なルールの整備や管理体制の構築を進めてきた。

⁶⁸ Tier1 のサプライヤに部品等を納入するサプライヤ

現在、OSS ライセンスに関する取組については、技術管理部門が全社事務局として統括しており、設計開発に関わるルールの一つとして、OSS ライセンスに関するルールを整備している。また、各事業部門の技術部には OSS 管理事務局、及び製品ごとに OSS 管理リーダー/担当者が設定されており、技術管理部門で定めたルール等を一体となって技術部内に展開している。

具体的な OSS 管理ルールとして、開発チームにて OSS を使用する際、技術部の OSS 管理事務局及び OSS 管理リーダーに申請を行い、必ず承認を得なくてはならない。申請書を作成する際は、技術管理部門が用意する商用ツールを用いて、開発チームにて OSS の使用状況を把握する。OSS 管理事務局及び OSS 管理リーダーは、全社の OSS ライセンスに関するルールにて定めている各開発段階で確認すべき事項や OSS ライセンスごとの使用可否に基づき、申請書に記載された OSS のライセンスに問題がないかをチェックする。また、承認された申請書はデータベースに格納され、OSS 管理事務局／管理リーダー及び技術管理部門から参照出来るようになっている（図 4.7-1）。

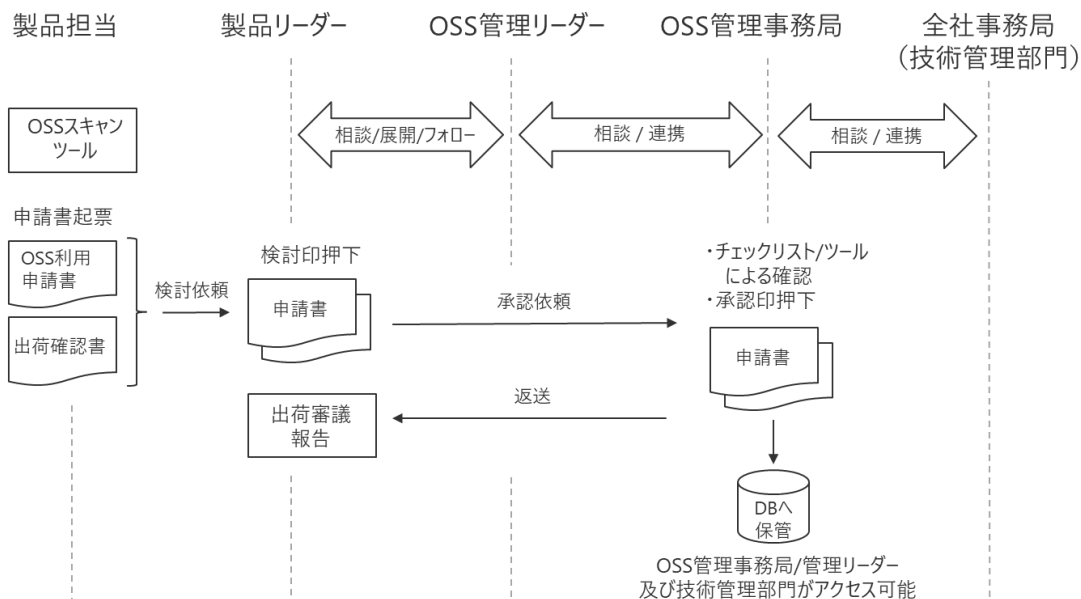


図 4.7-1 OSS 利用申請承認フロー⁶⁹

また、デンソーでは、自社の製品開発における OSS 使用だけではなく、サプライヤ（Tier2）から供給された部品等の中に OSS が使用されているケースも多々あり、それらに対しても OSS ライセンスのチェックが必要となる。ソースコードを入手できる場合には、前述の商用ツール等を用いてチェックできるものの、バイナリデータのみが納品されるケースが多い。したがって、契約等の取決めによって使用 OSS の情報提供

⁶⁹ デンソー提供資料を基に作成

を受けられるよう OSS の情報提供を盛り込んだ契約書の雛形を基に各サプライヤと調整を進めている。なお、サプライチェーンの下流（Tier2 以降）から上流（最終製品メーカー）において、ソフトウェア情報の共有を容易にする観点から、使用ソフトウェアリストのフォーマットを統一しており、フォーマットとしては OpenChain⁷⁰でトヨタ自動車等が利用を推進していることを理由に SPDX Lite を利用している。サプライヤ（Tier2）との使用 OSS の情報提供に係る調整は、時に難航することもあるものの、コンプライアンスの重要性の理解を得る等の工夫を図るとともに、各サプライヤが対応可能な範囲の妥協点を探り、取決め等の合意を丁寧に進めることで対応している。

<【脆弱性対応(v7)】脆弱性調査システムを用いた脆弱性対応>

OSS の脆弱性対応については、情報セキュリティ推進部門が運用する脆弱性調査システムを用いた取組を進めている。

具体的には、各事業部門の製品設計者に、その製品の使用 OSS のソフトウェア部品構成情報（SBOM）を作成し脆弱性調査システムへ登録してもらうことで、SBOM の情報をデータベース化し管理している。ソフトウェアの脆弱性情報は、NVD や JVN 等の情報を自動収集するほか、情報セキュリティ推進部門が Auto-ISAC や JPCERT/CC 等から収集し、脆弱性調査システムに連携している。脆弱性調査システムでは、これらの脆弱性情報と登録された SBOM の情報を照合し、一致度合の高い情報を製品の設計者に自動通知する。製品設計者は、情報の該否判断を行い、必要な対応を実施する。仮に、一定期間経っても対応が取られていないようであれば、脆弱性調査システムから自動で再度催促の通知を送る等の機能も備わっている（図 4.7-2）。

⁷⁰ OpenChain の活動概要については、トヨタ自動車の事例に記載

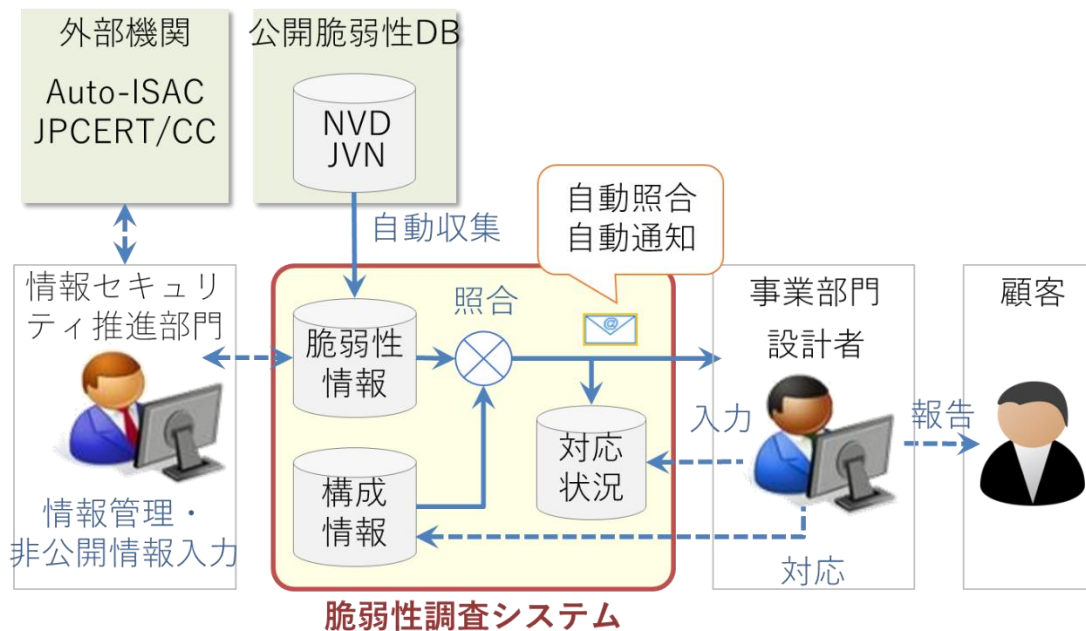


図 4.7-2 脆弱性調査システムの概要⁷¹

脆弱性対応では、一部商用ツールや外部リソースの活用もしており、情報セキュリティ推進部門では、各設計者がSBOMを作成する際、商用ツールを使用可能な環境を提供している。製品によっては多くのOSSを利用することもあり、そのような場合に、この商用ツールを用いてSBOMを作成することができるため、負荷を低減することが可能となる。

なお、OSSの名称づけは統一されていないため、収集したソフトウェア脆弱性情報が自社SBOMの情報と名称等において完全一致しない場合がある。機械照合では、アラートが出されるためには完全一致が必要となるため、このことが原因となり、うまく自動通知されない等の抜け漏れが発生する可能性がある。これを防ぐために、同一のOSSだが名称が異なるために別のソフトウェアだと判定されているものがないか、外部リソースを用いて手動で情報照合する形を採っている。

さらに、啓発・教育等の面においても工夫をしている。デンソーの製品群が高度の安全性を要求される自動車部品である特性上、各設計者は品質に対する高い意識を持っている。ソフトウェア脆弱性についても、品質問題として啓発することで、一連の取組に対する設計者のモチベーションを高めるようにしている。あわせて、OSSに係る対応は、通常の品質問題とは異なる知識が必要となるため、設計者に対して関連する教育プログラムも実施している。

⁷¹ デンソー提供資料を基に作成

デンソーでは、Tier2 以降のサプライヤが中小企業も多いために、いかに OSS 管理の負担を抑えながら運用ができるかを業界全体で検討する必要があると考えている。

ライセンス管理の面では、利用している商用の OSS 検出ツールによって検出される OSS の件数が多く、いかに効率的に管理をするかが課題である。

また、現在、脆弱性対応とライセンス管理は、必要な粒度が異なることや社内の取組の歴史的経緯から、別個に情報が管理されているものの、これらの共通化に向けて検討を始めている。その第一段階として、工数をかけずに SBOM を作成できるよう項目を統一する方向で議論している。

4.8. 富士通 ～部門横断の OSS 対応体制と全社統一的なソフトウェア管理～

■ 企業情報・本事例集における位置づけ

本社	東京都港区
業種	製造（電気機器）
従業員数	132,138 名（2019 年 3 月末現在）
商流	SIer 商流/SIer & 製品商流/最終製品メーカー
観点	「ライセンス」「サプライチェーン管理」「個の能力・教育」「組織体制」「コミュニティ活動」「選定評価」「脆弱性対応」「保守・品質保証」

■ 本事例のエッセンス

- OSS 対応体制を、OSS 対応組織のみならず、開発部門、管理部門等と密接に連携し、部門横断的に構築した。
- コミュニティ活動参加のルールを明確化するとともに、参加者がコミュニティに貢献できるような教育等を行っている。
- ソフトウェアライフサイクルの中で、全社統一的なツール、サービスを用いて、使用 OSS の見える化及びリスク検知を徹底している。
- ソフトウェア製品のセキュリティ確保に向けた取組において、OSS の使用状況管理と連携した対応を行っている。

■ 背景・課題

富士通は、国内大手の SIer、情報通信機器メーカーとして、常時 1000 以上の開発プロジェクトが進行しており、各プロジェクトでは OSS が多数使用されている。膨大な使用 OSS の管理を行うために、各フェーズにおいて、適切な対応を実施するのみならず、全社的な体制を整備し、社員のリテラシー向上を図るなど、総合的、多面的に対応をしなければならない。

■ 取組

<【サプライチェーン管理(S8)】【個の能力・教育(P8)】【組織体制(O8-1)】【コミュニティ活動(C8)】OSS 対

応に係る組織体制や教育体系の整備とコミュニティ活動＞

富士通では、OSS の専門組織として OSS 技術センター⁷²を設置しており、OSS の普及促進、技術支援、ガバナンス、リスクマネジメント、セキュリティといった多岐にわたる側面で取組を実施している。

一方で、全社的な OSS 対応体制としては、OSS 技術センターだけにとどまらず、開発部門、管理部門、社内コミュニティ⁷³とも連携するフォーメーションを組んでいる。具体的には、各部門、組織の専門知識を持った担当者が OSS の取組ごとに組織横断の対応プロジェクトを作り、課題解決に向けて定期的に活動を行っている。プロジェクトが一定の目的を達成した際には、既存の組織にその役割を引き継ぐ場合もある。

例えば、OSS のライセンスやセキュリティのリスクにおいては、管理部門の知財担当者、管理部門及び開発部門のセキュリティ担当者等がワーキンググループを作り、組織横断の体制で対応に当たっている（図 4.8-1）。その他、特定の OSS への対応強化を目的に、開発部門、SE 部門、OSS 技術センター等で実施している組織横断の活動等がある。

このような体制をとっているのは、OSS に係る問題解決が単一の知識領域のみで対応することが不可能なためである。中核となる組織、担当者を置きながらも、他部門の必要な組織、担当者と連携する部門横断の体制を築くことで、OSS リスクに対して高い対応力を発揮することを可能とした。

⁷² OSS ミドルウェアによるシステム構築から運用までを支援するため、2005 年 11 月に設置された。富士通グループや OSS ベンダをはじめ、コミュニティ、政府・OSS 団体と協力して OSS を活用したシステムのニーズに応える幅広い OSS ソリューションを提供するエンジニアが主体の組織である

⁷³ OSS の利活用促進、人材育成、支援体制拡充等の目的に基づき、社員主導で組織される複数の活動主体である

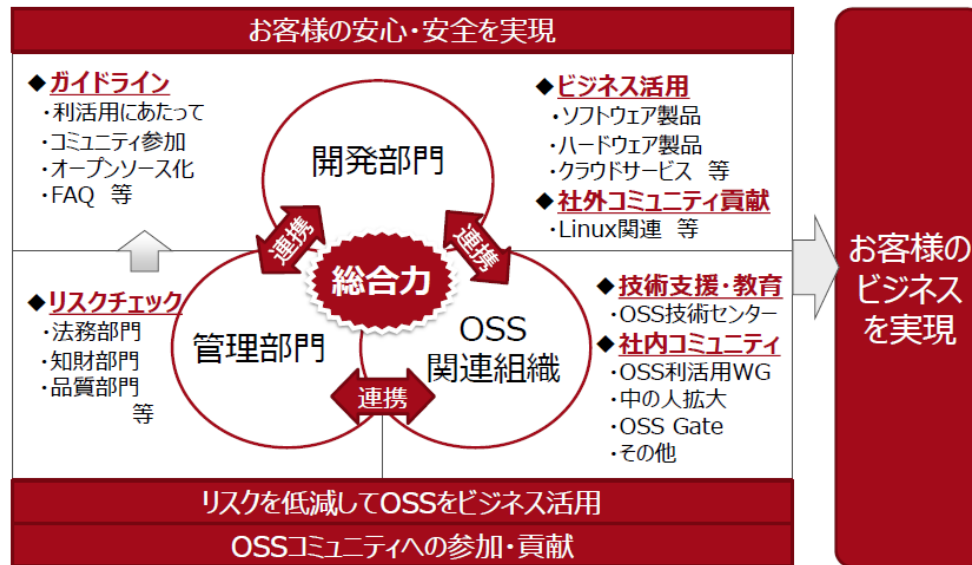


図 4.8-1 全社的な OSS 利活用を支援する体制・仕組みづくり⁷⁴

組織面での対応に加え、教育にも注力がされている。富士通では、OSS を利用するにあたり必要となる知識を学習する教育コースを、e ラーニングを通じて提供している。各分野を詳細に学べる技術者向けの講座をきめ細かく用意することに加え、OSS の知識は全社の DX 戦略と関係するため、OSS を含むシステムの利用者向けの講座も用意している。これらの e ラーニングに加え、OSS 開発者向けに、自身が OSS 開発者になったり、GitHub⁷⁵に参加する等の場面で必要となる知識についての教育も併せて実施している。

また、富士通は、外部のコミュニティ活動を OSS のビジネス利活用に貢献する有効な手段であると明確に位置づけている。しかしながら、社外活動であるために、社員のコミュニティ活動に対して適切なガバナンスを実施することが求められる。一例として、ある事業部門にとって有益であるコミュニティ活動が、別部門にとっては不利益となる可能性があるために、参加コミュニティを企業として管理する必要がある。したがって、富士通では、OSS コミュニティ参加ガイドラインを整備し、適正なコミュニティ活動が実施されるようコントロールしている（表 4.8-1）。

⁷⁴ 富士通提供資料より抜粋

⁷⁵ ソースコードをホスティングするソフトウェア開発のプラットフォームサービス

表 4.8-1 OSS コミュニティ参加ガイドラインの概要⁷⁶

目的	OSSコミュニティへプログラム等を投稿する際の社内手続き、ソフトウェア開発者が遵守すべき事項、及び社内関連部門の役割を明確にする
対象	富士通Gにて、既存のOSSに反映するためにOSSコミュニティへプログラムやドキュメント等の投稿を行う部門、及び開発者等の関係者
構成	<第1章 OSSコミュニティへの参加申請> ● コミュニティの評価 対象OSSのライセンスに不都合はないか、コミュニティ活動は活発か ● 申請手続き 社内Webから、法務部門の契約チェック、技術部門の承認を受ける ● 申請部門の運用決定 申請部門において、投稿先、投稿者、投稿手順等を決定 <第2章 プログラム投稿時の自己チェック> ● 著作権 他社の著作物やOSSを含んでいないか ● 秘密情報 自社、他社の秘密情報を含んでいないか ● 機能追加を行う場合 特許侵害はないか、特許出願は可能か ● 投稿 部門にて設定した手順で投稿。投稿管理は投稿部門の幹部社員の責任

また、コミュニティ参加者に対して、社外活動における情報漏洩等のリスクに関する研修に加えて、コミュニティに参加する動機付けやコミュニティ内でプレゼンス発揮可能となる研修等、コミュニティに貢献するための教育も行われている。これらの一連の取組により、コミュニティ活動の活性化とそれによる自社 OSS 利活用の発展が持続することを目指している。

参加しているコミュニティの具体例として、OpenChain⁷⁷があげられる。富士通は、OSS が商用代替ソフトから、業種を問わず市場で広く期待され、デファクトや標準を生み出すものへと役割を拡大する中、その潮流を掴み、将来動向の早期把握を実現すること等を目的に、2019 年 3 月にプラチナメンバーとして

⁷⁶ Japan Technical Jamboree 60（2017 年）資料を基に作成

<https://elinux.org/images/e/e7/%E3%80%90CEWG%E3%80%91OSS%E3%82%B3%E3%83%9F%E3%83%A5%E3%83%8B%E3%83%86%E3%82%A3%E5%8F%82%E5%8A%A0%E3%81%AB%E3%81%A4%E3%81%84%E3%81%A6.pdf>

⁷⁷ OpenChain の活動概要については、トヨタ自動車の事例に記載

OpenChain に加入した。具体的には、Japan WG⁷⁸、及び配下の Sub-WG（Tooling SWG⁷⁹、FAQ SWG⁸⁰、Planning SWG⁸¹）において議論を主導しているほか、2019 年 11 月には、OSS を管理するためのコンプライアンスの要件である OpenChain2.0 の「仕様」への適合を自己認証している⁸²。富士通では OSS を積極的に製品に組み込む部門等から認証取得を開始しており、特にサプライチェーン管理に係る部署に認証取得の輪を広げ、グループ会社でも認証が取得されることを目指し、取組を進めている。

<【選定評価(S8)】【ライセンス(L8)】【脆弱性対応(V8-1)】ソフトウェアのライフサイクル管理>

富士通では、ガイドラインや規約の整備を行い、各フェーズにおける OSS の適正使用がなされているかチェックするように定めている。以下は、プロジェクトにおいて OSS を適用する際のガイドライン（OSS 利活用ガイド）の構成である（図 4.8-2）。特に、第 2 章「各開発フェーズと目的」では、企画段階における適切な OSS 選定、開発の各フェーズにおけるライセンスのコンプライアンス徹底、出荷前の必要アクション実施に係るエビデンス保存及びサポート体制構築等の実施を義務付けている。

⁷⁸ OpenChain Japan WG の具体的な取組については、トヨタ自動車の事例に記載

⁷⁹ OSS コンプライアンス用のツールの検討や検証が行われる SWG

⁸⁰ OSS ライセンス関連の FAQ が共有される SWG

⁸¹ Japan WG の進め方についての議論が行われる SWG

⁸² OpenChain2.0 とほぼ同内容の OpenChain2.1 が ISO 化（ISO/IEC 5230:2020）されており、OpenChain2.0 に準拠した認証を取得している企業は、自動的に ISO/IEC 5230:2020 の要求事項に適合する。

<https://www.openchainproject.org/featured/2020/12/15/openchain-2-1-is-iso5230>

第1章：	OSS利活用ガイドについて（目的、対象、利用者、用語の定義）
第2章：	<u>開発フェーズ毎に必要なアクションと解説</u>
第3章：	<u>ライセンス条件の遵守のための留意事項</u> （調査方法、特徴、遵守ポイント）
第4章：	OSS関連サイト（各種問合せ先）

【第2章 各開発フェーズと目的】

1. 企画	プロジェクトの目的達成に必要な構成要素、条件、関係組織等を明確にし、 <u>利用可能なOSSの選定</u> 、関係組織との調整を行う。
2. 開発	基本設計、設計、プログラミング、試験(テスト)の4工程とし、OSS <u>ライセンスを遵守</u> し、当社知的財産を保護した開発を行う。
3. 出荷前	出荷判定や製品登録の際に実施する最終審査であり、利用OSSに関する必要なアクションの実施、 <u>EVIDENCEの保存</u> を確認する。
4. アフタ準備	製品、サービス、システム等の維持管理や顧客支援の為の <u>体制を準備</u> し、利用OSSの脆弱性等への対応の運用を明確にする。
5. アフタ（サポート）	事前に定めた体制を維持しながら、運用手順に従いOSSを含む製品、サービス等のサポートを実施する。

図 4.8-2 OSS 利活用ガイドの構成及び第 2 章の詳細⁸³

また、各プロジェクトにおける OSS の使用状況（使用 OSS、バージョン、ライセンス）等を全社的に管理している。具体的には、開発の各フェーズにおいて、管理部門や OSS 技術センター、さらには CSIRT⁸⁴ チームが連携し、以下のサービスをプロジェクトに提供している（図 4.8-3）。

- ① 検討・設計段階では、OSS のライセンス等に関する自己チェックツールを提供。
- ② 開発・導入段階では、OSS の混入を診断するサービスを提供。
- ③ 保守・運用段階では、機能面の重大バグを含む脆弱性や EOL⁸⁵を管理し、関連部署にアラートを通知するサービスを提供。

⁸³ 富士通提供資料より抜粋

⁸⁴ Computer Security Incident Response Team:コンピュータセキュリティインシデントに対応するための専門チーム

⁸⁵ End of Life

■ オープンソースの専門組織＋管理部門が主導する形で
安心・安全に利用する為の各種サービスを全工程にわたって提供。



図 4.8-3 OSS 管理の一例⁸⁶

あわせて、①におけるサービスとして、適切な OSS の選定をサポートすることを目的に、自社で使用するであろう OSS の開発傾向、ライセンス、過去に発生した脆弱性（傾向やクローズまでの期間等の情報）等をデータベース化して、社内向けに公開している。これらの各種のツール、サービスも規約等で使用が定められており、全社全てのプロジェクトへの適用が徹底されている。

以上のとおり、全社的にツール、サービスを提供し、企業として統一された運用を行うことにより、セキュリティ脆弱性リスクやライセンスリスクの低いソフトウェアライフサイクルを実現している。

<【脆弱性対応(v8-2)】【保守・品質保証(M8)】【組織体制(O8-2)】セキュリティ確保に向けた各種取組における OSS への対応>

富士通では、構築したシステムを顧客へ納入する前に、社内の独立したセキュリティ監査部門によるセキュリティ監査を実施している。システムに OSS を使用している場合には、前述の OSS 管理において用いられるツールと連携し、OSS のバージョン、当該バージョンに内在する脆弱性の有無、対応漏れの有無等を確認している。

⁸⁶ 富士通提供資料より抜粋

また、富士通は、OSS ベースのクラウドサービスを数多く提供しているため、クラウドサービス専門のセキュリティインシデント対応を目的とした、富士通クラウド CERT⁸⁷を組織している。OSS に起因するセキュリティインシデントが発生した場合、富士通クラウド CERT が OSS 技術センター等の社内関連部署と連携して、応急対応を行うこととなっている。

さらに、ソフトウェア製品に含まれる未検知の脆弱性を、AI により評価・検出する技術を用いた検証サービスを提供している⁸⁸。富士通の一部製品・サービスでは、この検証サービスを用いてセキュリティ品質を高める取組を開始している。

⁸⁷ Compute Emergency Response Team : セキュリティ、インターネット上の不正アクセス、ソフトウェアの脆弱性などの情報収集や情報提供を行う組織

⁸⁸ 「脆弱性検証サービス」を提供開始

<https://www.fujitsu.com/jp/group/fct/resources/news/press-releases/2016/1101.html>

4.9. 日本電気（NEC） ～事業部ごとの取組から全社的取組へ～

■ 企業情報・本事例集における位置づけ

本社	東京都港区
業種	製造（電気機器）
従業員数（連結）	112,638 人（2020 年 4 月末現在）
商流	SIer 商流/SIer & 製品商流/最終製品メーカー
観点	「選定評価」「ライセンス」「脆弱性対応」「個の能力・教育」「保守・品質保証」「組織体制」

■ 本事例のエッセンス

- 統一的なガイドライン、OSS 情報共有システム等を構築することにより、事業部ごとの OSS への取組を全社的な取組へと昇華させた。
- 全社の OSS 使用状況を一元管理し、企画から運用・保守の工程で必要な対策についてワンストップなサポートを実現するため、ツールやワークフローシステムを活用した「OSS 利活用プロセス」を構築した。
- OSS 利活用プロセスを全社ルール化して、トップダウンで適用を指示することで、適用プロジェクトを増やし、高い効果をあげた。
- PSIRT を設立し、脆弱性情報の収集、対応方針の整理、対応の社内調整を行っている。また、脆弱性管理を効率的に実施するためのシステムを開発し運用している。

■ 背景・課題

NEC では、国内大手の SIer として、常時多数の開発プロジェクトが進行している。一方で、ソフトウェアや情報通信機器等の製品提供、SaaS⁸⁹形式のサービス提供等も展開している。それら多様な価値提供を行う中において、それぞれの事業の状況に合致しつつも全社統一的である OSS の管理/対応が求められる。

自社製品や顧客システムの脆弱性管理を個別に対応するには、体制やコスト面で負荷が高い状況が続いている。また、NEC グループとして適切な脆弱性対応が求められる。

⁸⁹ Software as a Service : クラウドで提供されるソフトウェア

■ 取組

＜【選定評価(S9-1)】【ライセンス(L9-1)】【組織体制(O9-1)】事業部ごとの取組から全社的な取組へ＞

NEC では、2000 年代初頭から Linux サービジネスにおいて、OSS の利用を開始した。また、ライセンスに配慮する必要⁹⁰がある点から、主に組み込み系製品（携帯電話等）や B2B2C⁹¹の SI ビジネスにおいても、早くから事業部独自で OSS 管理の取組を実施していた。2010 年以降、OSS 利活用のメリットが認識され、その利用が増える中、事業部ごとであった OSS 利活用及び管理の取組を全社的なものに昇華させる必要性を認識した。そのため、2014 年にタスクフォースを結成し、全社的な OSS 管理の取組を開始した。現在では全社的な OSS 対応組織である先端 SI 技術開発本部 OSS 推進センター（以下、「OSS 推進センター」という。）が事業横断的に OSS 利活用及び管理を推進している。これまでに実施してきた主な取組は以下のとおりである。

- ① ガイドラインの整備
- ② 推奨 OSS リスト等の整備、非推奨 OSS のリスク調査
- ③ OSS 情報共有システムの整備
- ④ OSS コード検証、OSS ライセンス遵守支援

以下、各取組の詳細について記載する。

① ガイドラインの整備

OSS 使用の際の遵守事項について、ソフトウェア関連プロジェクトの各フェーズにて参照するガイドライン群を段階的に整備した（図 4.9-1）。一例として、OSS 使用判断ガイドラインでは、企画や開発フェーズにおける OSS の導入可否やライセンス検討の基準を定めている。各ガイドラインは個別に定義されたものであるため、全体がより統一的なものとなるよう継続的に見直しが行われている。また、契約書雛形等の共有も実施している。

⁹⁰ 製品提供によってソフトウェア（OSS）が再配布されることからライセンスを遵守する必要がある。

⁹¹ Business to Business to Consumer：企業が他の企業へ商品やサービスを提供し、提供を受けた側の企業がさらに一般消費者へ提供する商取引の形態

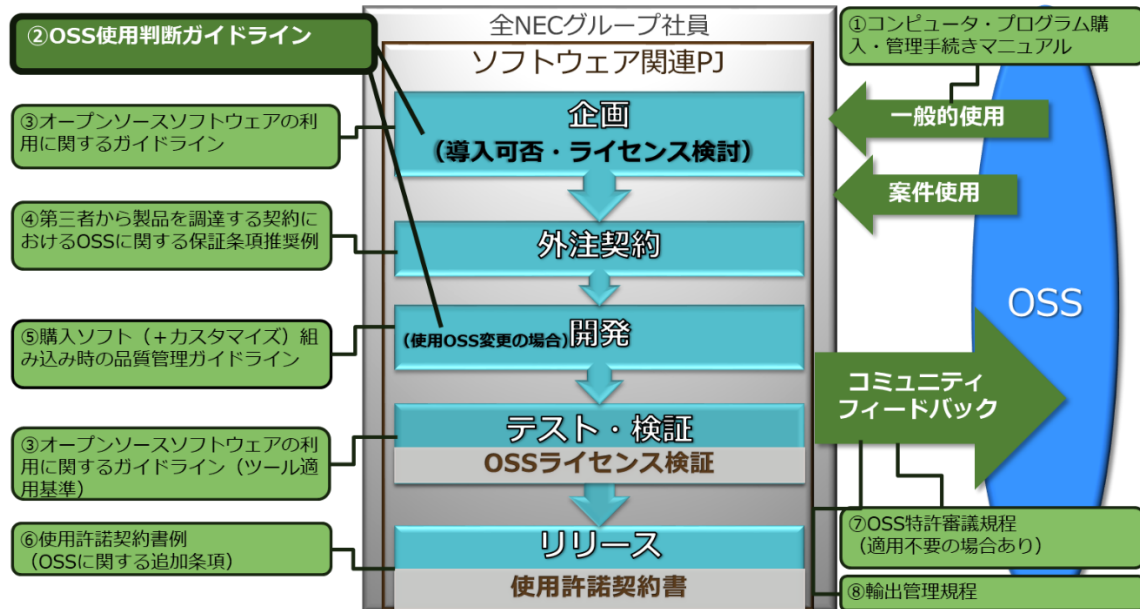


図 4.9-1 ソフトウェア関連プロジェクトの各フェーズにて参照するガイドライン群⁹²

② 推奨 OSS リストの整備、非推奨 OSS のリスク調査

使用を推奨する OSS をリスト化し、社内提供することで、後々のリスク（ライセンス違反、脆弱性、EOL 等）を大幅に低減させている。使用を推奨する OSS はサポートの有無、サポート実施主体、社内の利用実績に基づいて判断している。あわせて、「ハイリスク」「非推奨」のリストも整備し、使用を控えるべき OSS も明確化している⁹³。重大な脆弱性がある、もしくは EOL の場合は「ハイリスク」、サポートが完全でない場合は「非推奨」に分類している。また、非推奨である OSS についても、要望や質問を受けた場合は OSS 推進センターが調査を行い、リスク回避のアドバイス等を実施している。

③ OSS 情報共有システムの整備

②の推奨 OSS リストの他、各ソフトウェア分野⁹⁴のメジャーな OSS に関する公知情報⁹⁵、及び当該 OSS の自社内使用実績に基づく情報（サポート体制等）を Web ポータル（情報共有システム）上で提供している。それらの情報は OSS 推進センターが定期的にアップデートしている。

⁹² NEC 提供資料より抜粋

⁹³ 各リストは、定期的（数か月ごと）に更新をしている。

⁹⁴ 日本 OSS 推進フォーラム作成の OSS 鳥瞰図から、各ソフトウェア分野を参照している。

⁹⁵ Black Duck Open Hub、JVN iPedia、各 OSS のホームページ等を情報源に、Version、規模、言語、ライセンス、脆弱性、Activity の情報を収集している。

④ OSS コード検証、OSS ライセンス遵守支援

実際の製品・システムに、外部流用等によって混入した認識外の OSS が含まれていないか、商用ツールによって検出するサービスを社内提供している。また、ドキュメント作成、ソースコード開示等、OSS ライセンスに対する適切な対応・遵守を支援する相談窓口を設置している。

<【選定評価(S9-2)】【ライセンス(L9-2)】【組織体制(O9-2)】網羅的な OSS 管理の実現>

上述のように、NEC では OSS の利活用及びその管理について全社的な取組を進めてきた。現在では、アジャイルや DevSecOps⁹⁶の開発スピードへの対応における課題意識から、より網羅的に OSS を管理するため「OSS 利活用プロセス」を構築している（図 4.9-2）。これは、全社の OSS の使用状況を OSS 推進センターが把握することで、同センターによるワンストップなサポートを実現するものである。OSS 利活用プロセスの構築に当たっては、ソフトウェア関連プロジェクトの既存プロセスに新たな対応を追加する形が採られており、適用対象は NEC の製品・サービス・SI 全般に渡っている。

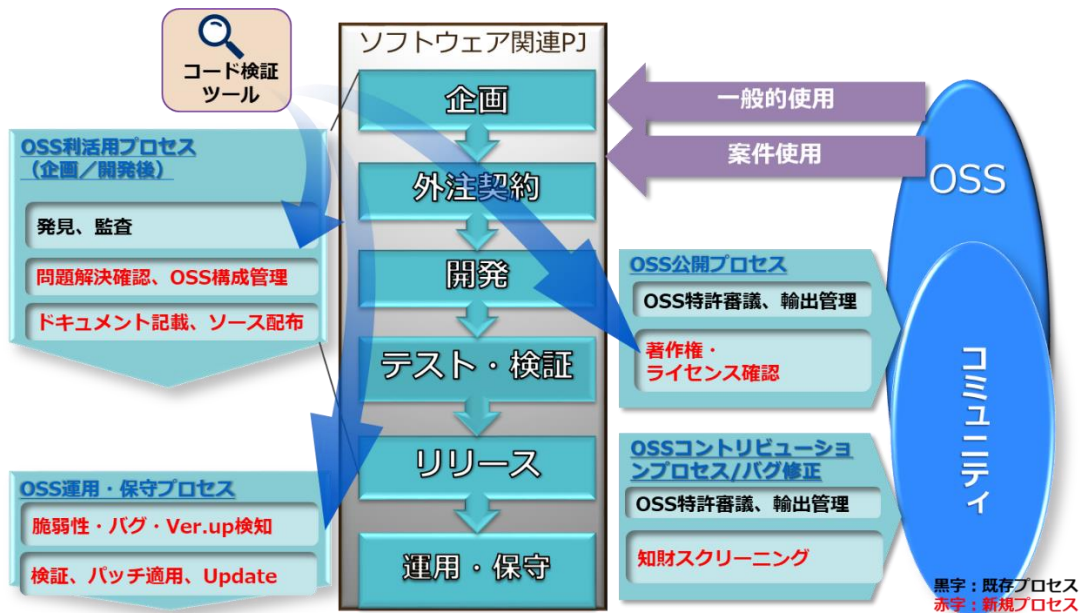


図 4.9-2 OSS 利活用プロセスの全体像⁹⁷

OSS 利活用プロセスはワークフローシステム上で進行する。一例として、開発後の OSS ライセンス遵守

⁹⁶ アプリケーションとインフラストラクチャのセキュリティを、開始時点から考慮すること

⁹⁷ NEC 提供資料より抜粋

に関する流れは、以下のとおりである。

- ① 使用 OSS リストのワークフローシステムへの取り込み、及び OSS リスク情報の提供
- ② OSS リスクへの対処、及び OSS 構成情報の記録
- ③ OSS ライセンス遵守のフォロー
- ④ 開示する OSS コード等の管理

以下、各取組の詳細について記載する。

- ① 使用 OSS リストのワークフローシステムへの取り込み、及び OSS リスク情報の提供

プロジェクトで使用する OSS は、申請（OSS リスク管理票の提出）、もしくはコード検証ツールによる OSS 検知を起点にワークフローシステムに取り込まれる。OSS 推進センターはプロジェクトに対して、使用する OSS に関するリスク等の情報提供を行う。

- ② OSS リスクへの対処、及び OSS 構成情報の記録

OSS にリスク等がある場合は、プロジェクトにおいて対処方法を決定し、それを OSS 推進センターが確認する。確認の結果が問題ない場合、もしくは元々リスク等がない場合は、OSS 推進センターがソフトウェア構成情報を記録し一元管理する。

- ③ OSS ライセンス遵守のフォロー

OSS ライセンスを遵守する上で必要となる通知文等の作成を OSS 推進センターが支援する。また、プロジェクトでは開示を要するソースコードを準備する。OSS 推進センターにてそれらを最終確認した後、OSS とともに頒布、または所定の掲載場所に公開する。

- ④ 開示する OSS コード等の管理

OSS 推進センターにて、頒布または公開したソースコード等の管理をフォローする。

この他にも、コード検証ツールの機能を用いて、脆弱性、バグ、バージョンアップ情報等を各プロジェクトが確認でき、必要に応じ OSS 推進センターがその対応を促す運用フローも OSS 利活用プロセスには含まれている。

また、当初、OSS 利活用プロセスは、各プロジェクトへの「お願い」ベースで草の根的に展開していたものの、その効果を高めるため、OSS 利活用プロセス適用を全社ルールとして定め、経営層が参加する会議においても周知した。全社ルール化とトップダウンの指示により、社内における OSS 利活用プロセス適用に対する意識が高まり、適用件数は増加傾向にある。現在、OSS 利活用プロセスの運用によって得られた知見を次の開発に活かす PDCA サイクルを構築する目的で、利用実績の多い OSS や組み合わせて利用される OSS などの情報を、推奨 OSS リストや次世代 OSS 情報共有システムへフィードバックする仕組

みづくりを検討している（図 4.9-3）。

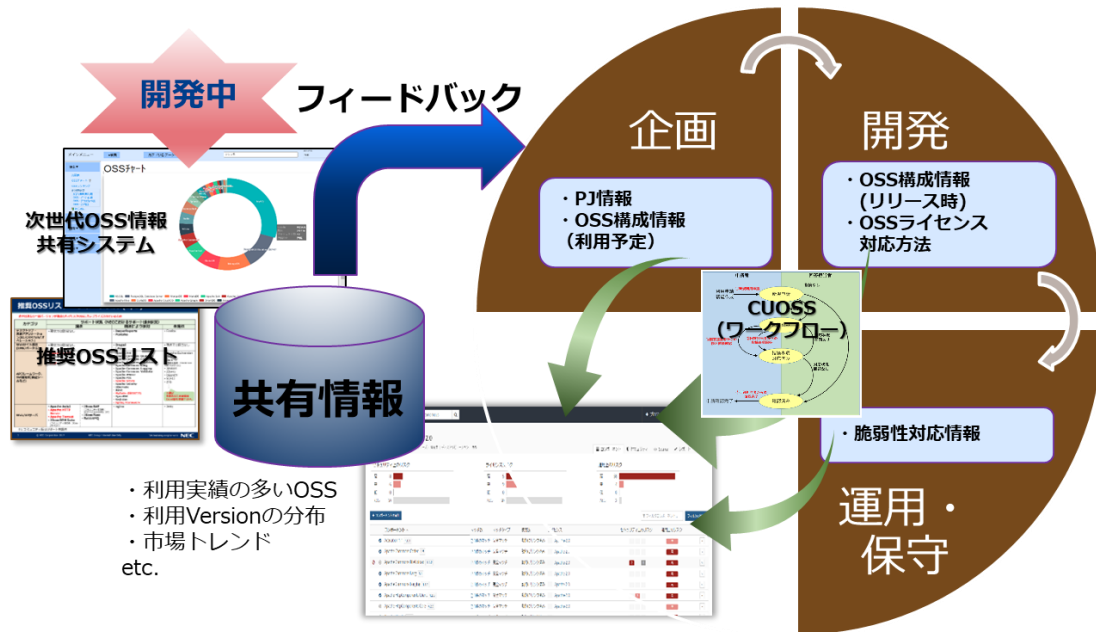


図 4.9-3 OSS 利活用プロセスの PDCA サイクル（検討例）⁹⁸

今後の取組として、新たなツールの PoC⁹⁹等も検討している。また、各ツール独自の OSS 名前マスターを統一することや各種フローの自動化等、効率化の策も検討しつつ、社内の OSS に対する更なる意識向上等、多面的に取組を進めている。

<【脆弱性対応(v9-1)】【個の能力・教育(p9)】PSIRT の設立>

NEC グループでは、2004 年に PSIRT を設立した。当時、Code Red/Nimda などの脅威、SQL インジェクションなど経営を脅かすサイバー攻撃が増加していた。また、自社製品や顧客システムの脆弱性管理を担当者が個別に対応することに、体制やコスト面でも限界を感じていた。そこで、専門組織を設立し、仕組み開発（後述する VPS などを参照）に着手することになった。また、PSIRT 設立当初は参考となる情報がなく手探り状態であったが、FIRST の PSIRT Services Framework¹⁰⁰ が公開された後は、PSIRT 構築や

⁹⁸ NEC 提供資料より抜粋

⁹⁹ Proof of Concept: 新しい概念や理論、原理、アイデアの実証を目的とした、試作開発の前段階における検証やデモンストレーション

¹⁰⁰ 製品やサービスを開発し提供する組織が PSIRT を設置し、継続的に運用し、その能力の向上を支援するためのガイド、FIRST publishes Security Incident Response Teams (SIRTs) Services Framework Version 1.0, 30 Mar 2016、Press Releases 2016、

対応内容等において参考情報として活用した。

PSIRT は、JPCERT/CC、外部ベンダ、個人などの窓口となり、脆弱性情報（公開済み情報と未公開情報を含む）を取得している。取得した脆弱性情報を基に、影響する製品、対応方針を整理し、関係する開発側の責任者と脆弱性の調査や対処を行う。

また、脆弱性の早期警戒情報やインディケータ情報の収集および社内関係者への発信、世の中で流行している脆弱性の注意喚起やレポートの作成、社内公開も行っている。

加えて、インシデント発生時に説明責任を果たせるよう、NEC グループ製品の脆弱性を発見した際の報告や対処方法について PSIRT が一元的に相談を受け付けている。NEC グループとして適切な脆弱性対応を行うように取組んでいる。

<【脆弱性対応(v9-2)】【保守・品質保証(M9)】脆弱性予防管理システム（VPS）>

PSIRT では、脆弱性予防管理システムである Vulnerability Preventive management System（VPS）を構築している。VPS は、開発するシステムの構成情報を登録し、構成情報とともに脆弱性対策を管理するシステムである。発見・公開された脆弱性情報を収集し、登録されている構成情報の何れかに影響する場合に社内の関係者へ脆弱性情報を配信する仕組みである。配信後、対応が漏れないよう、対処予定情報を VPS に登録しておき、実際の対応状況を月次で事業責任者へ報告する。なお、脆弱性情報の収集については、NEC の関連会社である NEC ソリューションイノベータの脆弱性情報収集サービスを利用しており、日本語訳された脆弱性情報がタイムリー（毎日 1 回以上配信）に連携されている。

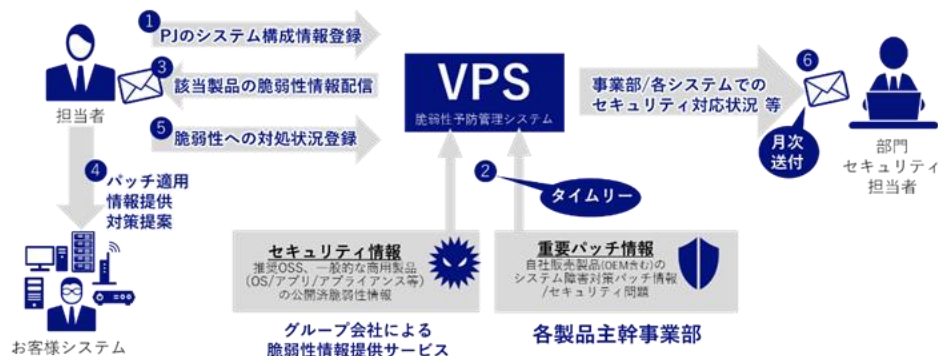


図 4.9-4 脆弱性予防管理システム（VPS）のしくみ

2006 年頃からサービス開始・段階的な機能拡張を行い、2011 年からは現在の仕組みで図 4.9-4 のとおり稼働している。具体的には、利用する OSS 等の構成情報入力・管理機能、脆弱性情報の収集・閲覧機能、脆弱性情報のマッチング及び配信機能、パッチなど対応・適用判断の入力機能、事業責任者への定期報告機能を追加した。

<<https://www.first.org/newsroom/releases/2016>>

VPS へ事前に登録するシステムの構成情報は、OS、ミドルウェア、モジュール等の粒度となっている。その理由は、VPS は人が直接管理・運用することを前提としており、最低限管理すべき脆弱性対象の粒度として OS、ミドルウェア、モジュール等の単位にフォーカスしているためである。なお、製品やプロジェクトによって、製品やプロジェクトの担当者から特定コンポーネントの脆弱性情報配信や構成情報管理希望があった場合は、より詳細な単位であるコンポーネントを管理することになっている。

VPS で登録されている製品と脆弱性情報のマッチングは、製品情報のソフトウェア名とバージョンを基に、脆弱性情報を一覧から検索する。VPS の良い点は、日本国内の拠点で使用する場合、脆弱性情報が日本語であるため、現場レベルでの理解が容易なことである。海外ベンダの構成管理ツールの場合、コンポーネントレベルまでの把握が可能であっても、脆弱性情報が英語表記であるため、現場レベルでの理解は難しい。そのため、NEC としては、VPS と海外ベンダの構成管理ツールを併用している。

現在も機能拡張は継続しており、直近では、VPS への構成情報や対処状況データを取得し VPS へ登録する作業負荷が高い点を解消するため、構成情報を収集するスクリプトを配布し、構成情報を登録代行するサービスを導入した。今後は、出荷前に行う脆弱性スキャンの結果等を VPS に取り込むことで、DevOps¹⁰¹の考え方を反映することも検討している。

¹⁰¹ ソフトウェアの開発担当と運用担当が協力する開発手法

4.10. 日本電信電話（NTT） ～OSS サポートに係る適切な役割分担～

■ 企業情報・本事例集における位置づけ

本社	東京都千代田区
業種	情報通信
従業員数（連結）	303,350 名（2019 年 3 月末時点）
商流	SIer 商流/SIer
観点	「脆弱性対応」「組織体制」

■ 本事例のエッセンス

- 各グループ企業の OSS 対応組織等を適切に役割分担した上で、OSS に係る対応体制を整備した。

■ 背景・課題

NTT は、国内大手の情報通信企業として、多数のグループ企業を有している。NTT、各グループ企業双方で、常時多くの開発プロジェクトが進行しており、各プロジェクトでは OSS が多数使用されている。OSS に係る対応は、企業の枠を超えて、適切な役割分担の下、グループとして一体となった取組を行う必要がある。

■ 取組

<【脆弱性対応(v10)】【組織体制(o10)】OSS 対応に係る体制の整備>

NTT では、OSS に係るサポートの専門組織として、NTT OSS センタ（以下、「OSS センタ」という。）を設置している。プロジェクトの設計、開発、運用における OSS 問い合わせ対応、トラブルシューティングサービス、技術検証済の OSS 組み合わせパターンの整理（OSSVERT¹⁰²）、情報提供サービス等を実施している。また、OSS センタは、研究機能も有しており、革新的 ICT サービスの創生及びソリューションの拡大もミッションとし、各種コミュニティにおける開発連携等も積極的に行っている。

OSS センタは、NTT グループ企業に対して、それぞれの企業のニーズに合わせたサービスを提供してい

¹⁰² OSs Suites VERified Technically：複数の OSS 等を組み合わせる場合に、その性能や適切な設定を動作検証し、参照情報として技術資料に取りまとめる活動、及びその情報提供

る。一例として、ソフトウェアライフサイクル管理の取組において、OSS センタは、NTT 本体には各グループ企業へガバナンスを効かせる目的で保守期間が終了したソフトウェアのリストを提供する一方、各グループ企業には OSS のバージョンアップ・移行等の現場支援を行い、連動性がありつつも各企業に即した対応等を行っている。

さらに、OSS センタの問い合わせ窓口であるポータルサイト上では、まだサポート対象になっていない OSS に関しても、旬であれば積極的に技術情報を発信している。要望があれば技術支援を行うとともに、情報提供ページのアクセス状況を分析し、OSS センタからアプローチする取組も行っている。進化の速い OSS 市場に対応した最適なサービスを提供し、各グループ企業のニーズに俊敏に応えることで OSS の利活用を促している。

また、OSS センタは、サポート面においては NTT の研究部門やサポートベンダと連携するとともに、OSS 関連の脆弱性が発生した場合は、NTT の技術部門やグループ内 CSIRT（NTT-CERT）と連携して対応する等、他部門との協力関係も構築している。具体的な一例として、前述のとおり、OSS センタのメンバーが様々なコミュニティで活動しているために、最新の脆弱性情報やセキュリティ関連情報をいち早く入手する機会がある。その場合、OSS センタが NTT-CERT の脅威情報配信のスキームに参画して当該情報を共有している。

OSS センタの活動概要・位置付け及び NTT グループのセキュリティ脅威情報配信スキームへの参画状況は以下のとおりである（図 4.10-1、図 4.10-2）。

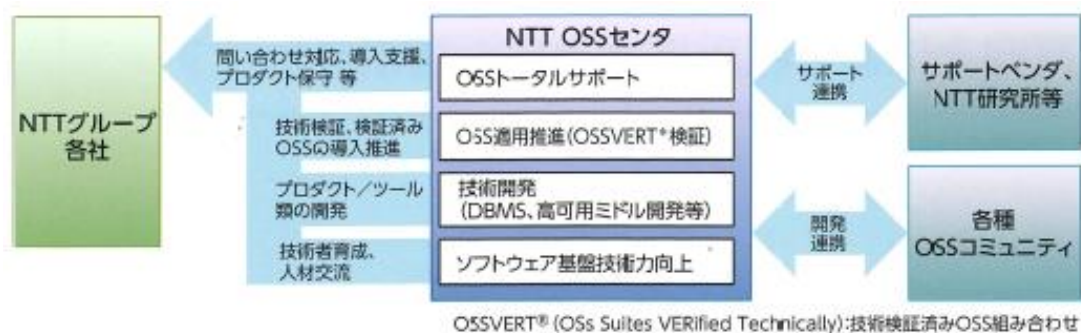


図 4.10-1 OSS センタの活動概要・位置付け¹⁰³

¹⁰³ NTT 提供資料より抜粋

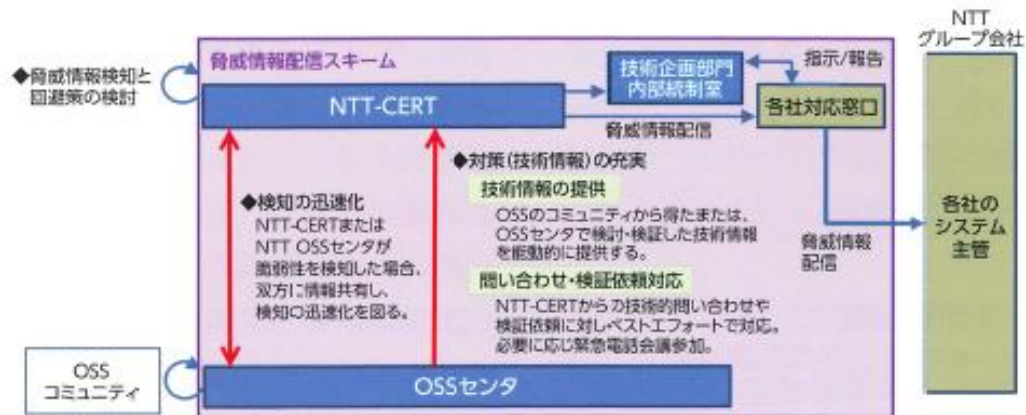


図 4.10-2 OSS センタの NTT グループのセキュリティ脅威情報配信スキームへの参画状況¹⁰⁴

一方で、各グループ企業においても、それぞれ OSS 対応組織が存在している。OSS センタと各グループ企業の OSS 対応組織は、協力関係にあり、適切な役割分担の下にそれぞれの対応を行っている。例えば、サポート業務において、OSS センタは、各グループ企業の OSS 対応組織がエスカレーション¹⁰⁵してきた問い合わせ対応等を行っている。

¹⁰⁴ NTT 提供資料より抜粋

¹⁰⁵ 各グループ企業の OSS 対応組織はその問い合わせの内容に応じて、自組織対応をするか、OSS センタへエスカレーションするかを決めている。

4.11. 匿名企業 A 社 ～OSS 選定基準の明確化とコミュニティ活動～

■ 企業情報・本事例集における位置づけ

本社	－
業種	情報通信
従業員数	－
商流	SIer 商流/SIer
観点	「選定評価」「保守・品質保証」「コミュニティ活動」

■ 本事例のエッセンス

- ・ 使用を推奨する OSS リストと使用を禁止する OSS リストを整備し、OSS 選定基準を明確化している。
- ・ コミュニティ活動を自社業務の範囲内として捉え、参加を全面的に支援している。

■ 背景・課題

A 社の開発プロジェクトでは OSS が多数使用されている。そのため、OSS に係る対応は全社統一的なルールの下実施される必要がある。

■ 取組

<【選定評価(S11)】【保守・品質保証(M11)】全社統一的な OSS 選定>

A 社は、OSS のリスクマネジメントにおいて、OSS 選定フェーズが最も重要であると考えている。具体的には、使用を推奨する OSS リストと使用を禁止する OSS リストを整備し、社内共有している。この内、使用を推奨する OSS の選定基準は概ね以下のとおりである。

- ① 商流にディストリビュータ等を介しているために、サポート等の対応が担保されている。
- ② 当該 OSS のコミュニティ活動が活性化している。

③ A 社内の技術者がソースへの対応が可能で、ワークアラウンド¹⁰⁶を行える。

各リストは、定期的に審議の場が設けられており、その場において対象 OSS の追加や削除等の更新が行われている。各リストは A 社内において、OSS を選定する上での基準となっている¹⁰⁷。なお、A 社の OSS 対応部署¹⁰⁸においては、使用を推奨する OSS リストに記載されている OSS のみをサポートする形¹⁰⁹をとっている。

A 社における OSS に対する長い取組の歴史も相まって、これらのリストは社内で高い信頼性を持っている。これらの取組が一因となり、A 社内においては、OSS 利活用に対する抵抗感は低い。

<【コミュニティ活動(C11)】コミュニティ活動の支援>

A 社は、コミュニティ活動に多数参加している。コミュニティの活性化や自社ソフトウェアの OSS 化が A 社のビジネスにおける OSS 利活用に有効であるとの考えから、A 社はコミュニティ活動を自社業務の範囲内として捉え、参加を全面的に支援している。

A 社では、コミュニティ活動に関する内部発表の場や表彰制度を設けることで、その啓発も行っている。また、各コミュニティ活動におけるエキスパート社員が、コミュニティ活動へ若手社員等と一緒に参加することで、OJT による開発スキル等の伝承を行っている。

¹⁰⁶ 迂回措置、一時対処

¹⁰⁷ クライアント要望等の都合により、これらのリストに準拠しない OSS を使用する場合もある。

¹⁰⁸ システム開発及び運用の経験や、研究部門におけるソフトウェア研究開発の経験を持つエンジニアが主となる部署である。

¹⁰⁹ 例外として、ソフトウェアの種類やプロジェクトの状況に応じ、サポート対応する場合もある。その際は、サポートベンダとの連携やコミュニティの情報活用、ソースコードを確認すること等で問題解決を図る。

4.12. 匿名企業 B 社 ～グループ内 SIer を中心としたセキュリティ強化～

■ 企業情報・本事例集における位置づけ

本社	－
業種	情報通信（化学メーカー子会社の SIer）
従業員数	－
商流	SIer 商流/SIer & エンドユーザー（化学メーカー）
観点	「個の能力・教育」「脆弱性対応」

■ 本事例のエッセンス

- B 社が、グループ各社と深く連携しつつ、一体となったセキュリティ強化策を推進している。
- 使用 OSS を一元的に管理する体制の導入に向け、検討を進めている。

■ 背景・課題

B 社は、化学メーカー子会社の SIer であり、化学プラント等を現場に持つプロセス産業に対して IT ソリューションを提供している。

■ 取組

<【個の能力・教育(P12)】グループ内 SIer を中心としたセキュリティ強化>

B 社は、某化学メーカー及びそのグループ内における IT 機能の中核を担う会社であり、グループ各社に対する IT ソリューション提供や、システム管理を担う中で、セキュリティ強化も推進している。一例として、親会社である化学メーカーにおいては、情報システム部門に B 社社員が出向し、エンドポイントからサーバサイドまで、あらゆる領域でセキュリティに関する取組を一体となって行っている。そのようなセキュリティに対する活動の一環として、OSS セキュリティに関する取組も進めている。

後述の OSS の使用状況の管理に係る一連の対応においては、ソフトウェア脆弱性情報の正確性の確認や具体的対策検討及び実施等、人の判断を介する場面が多くある。そのため、グループ内においてセキュリティ強化の中核を担う B 社社員には、OSS のみならずセキュリティに関する高いリテラシーが必要不

可欠となる。B 社は、セキュリティに関する Web 教育体系を 10 年以上前から整備し、人材教育に力を入れており、試験で一定の得点を取るまで社員へ教育を行う等、リテラシーの底上げを図っている。なお、親会社がグローバル展開している拠点の IT 関連業務の担当者に対しても、B 社社員と同等レベルの教育を行うことを目指している。

<【脆弱性対応(v12)】OSS の使用状況の一元管理に向けた検討>

B 社の親会社である化学メーカー及びそのグループ内では、オフィスや化学プラント等において使用されるシステム（生産に関わるクリティカルなシステム含む）において、多くの OSS が使用されている。それらのシステム管理を担う B 社においては、現在、アプリケーションを所管する事業部とシステムインフラを所管する事業部が、それぞれにアプリケーション単位、サービス単位で OSS の使用状況を管理している。脆弱性情報の収集もそれぞれの事業部において手動で行われており、IPA やコミュニティ、Black Duck Open Hub 等が発信するソフトウェア脆弱性情報を定常的に収集している。必要な場合にはセキュリティパッチを適用する等の対応を実施しているが、あまりに対応が頻繁であると、アプリケーションのメンテナンス調整が難しいため、緊急の対応が必要でない限り、年 1 回程度を目安にセキュリティパッチの適用を含むマイナーバージョンアップを行うこととしている。

近年、パッケージマネージャー等の機能が発展し、有用なものとなった結果、OSS に開発者が認識していない OSS コンポーネントが含まれてしまうことがある。そのため、B 社では、OSS の使用状況を全社的に可視化できていないとの問題意識が高まり、約 2 年前から、バージョン、ライセンス、脆弱性を含めた OSS 使用状況の一元管理に向けて検討を開始した。

OSS の使用状況の一元管理を可能とする商用ツールの検討を進めたところ、ツールにより、アプリケーションで使用されている OSS や、当該 OSS ライセンスや脆弱性情報の把握までは自動化できるとの認識が得られた。一方、発見された脆弱性に対し、ツールが複数の対応方法を提示する場合があります、対応方法の選定に際して人の判断が必要となることが明らかになった。

この検討結果を踏まえ、B 社のセキュリティ所管部門が、商用ツールによって、各アプリケーション、サービスの OSS 使用状況についてのデータを一元管理しつつ、脆弱性情報取得時等にはそのデータをもとに各アプリケーションのチームが対応し、その対応の妥当性についてセキュリティ所管部門が精査するという体制を目指している。

4.13. 損害保険ジャパン ～ソフトウェア部品構成表を活用した脆弱性管理～

■ 企業情報・本事例集における位置づけ

本社	東京都新宿区
業種	金融（保険業）
従業員数	24,689 人（2020 年 4 月末現在）
商流	SIer 商流/システムエンドユーザー（金融）
観点	「個の能力・教育」「ライセンス」「脆弱性対応」

■ 本事例のエッセンス

- ・ ソフトウェア部品構成表（SBOM）を活用し、脆弱性やライセンス等について、抜け漏れのないリスク管理を実施している。
- ・ 経営層がサイバーセキュリティの重要性を強く認識しており、脆弱性対応においてトップダウンの啓発を行う等、必要性を理解してもらう工夫を行っている。

■ 背景・課題

損害保険ジャパンでは、多様化する金融サービスへのニーズに対応するため、2015 年より、システム子会社とともに、次期基幹系システムの刷新、及び様々な側面でデジタルトランスフォーメーション（DX）を推進している。これらを進める上では、OSS 等を含む広範な技術を取り扱う必要があり、また、業界の特性上、クリティカルなサービスレベルが要求されるため、そのシステム開発/運用において高度なソフトウェア管理が必要である。

■ 取組

＜【個の能力・教育(P13)】CIO を起点としたサイバーセキュリティに対する取組＞

損害保険ジャパンでは、CIO が起点となり、サイバーセキュリティへの取組を推進している。具体的には、CIO 自ら国内外の先進企業、IT ベンダ等からの情報収集を実施するとともに、社外人材の採用や人材教育等、サイバーセキュリティ体制の整備にも継続的に関与している。

その中で、損害保険ジャパンの OSS 対応組織¹¹⁰では、システム関連子会社と連携しつつ、スキルのある人材の登用等を通じてソフトウェア管理や脆弱性管理を強化している。特にシステム関連子会社においては、ソフトウェア管理専任組織を構築しており、変化の激しい DX に追従するための体制を組んでいる。

一方で、現場（システム運用担当者）レベルでは、急速に進む DX への対応等に伴う業務繁忙により、セキュリティに対する優先度が相対的に低下し、システムに脆弱性等による影響が出る可能性があったとしても、そのリスクを正しく認識してもらえない場合がある。このような状況を防ぐために、CIO 等からトップダウンで啓発を行う等、危機意識を高める工夫も行っている。

また、2018 年には、イスラエルのテルアビブにデジタルラボを立ち上げ、サイバーセキュリティに関するソリューションやノウハウを収集している。

<【ライセンス(L13)】【脆弱性対応(V13)】ソフトウェア部品構成表（SBOM）の活用>

損害保険ジャパンでは、近年構築したシステムにおいて、内製、ベンダ等から納品を受ける場合を問わず、多数の OSS モジュールが利用されている。その中で、OSS の脆弱性管理やリスク管理の観点から、ソフトウェア部品構成表（SBOM）を用いた OSS 管理を徹底している。具体的にはシステムの「開発前」「開発中」「最終納品及び運用」の 3 つのフェーズに分けて管理を行っている。

- ① 開発前のフェーズでは、調達管理に組み込む形で、申請ベースにて OSS 使用状況の大枠を把握する。
- ② 開発中のフェーズでは、工程管理に組み込む形で、スキャンツールを用いた OSS 使用状況の詳細把握を行い、SBOM を作成する。
- ③ 最終納品及び運用のフェーズでは、把握した SBOM をシステムごとにデータベースに蓄積し、脆弱性管理、リスク管理に活用する。

以下、各フェーズの詳細について記載する。

①開発前

各プロジェクトからの申請に基づき、OSS の使用状況（利用箇所・バージョン等）を把握する。また、その申請内容を基に「バージョン」「品質（脆弱性・コミュニティの活動状況）」「輸出管理の要否¹¹¹」「ライセンス（義務の発生有無）」といった観点でリスク分析及び評価を行う（図 4.13-1）。さ

¹¹⁰ 主に社内の情報セキュリティに対して、リスクを把握、管理することが主なタスクであり、ソフトウェア利用に係る戦略策定等を行っている。

¹¹¹ OSS をシステムに利用する際、それらのシステムが今後海外利用されるかを確認し、海外利用される場

らに、その評価とシステムの重要度¹¹²を踏まえ、必要に応じて対応策を検討する。これは開発前の段階で把握できるライセンス問題等のリスクに対して早期に対処する意図がある。この一連の業務は、調達フローに組み込まれる形で管理されている。

NO	OSS名	概要	利用サービス ※()内はバージョン				リスク分類			
			サービスA	サービスB	サービスC	サービス4	利用バージョン	品質 (脆弱性等)	輸出管理	ライセンス (義務の有無)
1	△△	プログラミング言語○○でソフトウェア開発を行う際に必要なソフトウェアを1つにまとめたパッケージ	○ (X123)				✕			
				○ (X456)				✕		
					○ (X789)					
						○ (X789)				✕
利用箇所・バージョンの可視化							リスク分析			

図 4.13-1 開発前の申請時に作成する OSS リスクに係るチェックリストフォーマット¹¹³

②開発中

開発段階では、実際のソフトウェアに対して、商用ツールを用いて、その OSS 使用状況を把握する。この過程において、①の際に把握しきれなかった OSS 情報の取得をするとともに、静的解析によるリスク検出を行う。損害保険ジャパンでは、これら取得した OSS 情報を SBOM として利用している。ツール上では、SBOM を用いて、ライブラリ単位で OSS のリスク及び脆弱性を可視化できる。アプリケーションごとにそれらのリスク等の解析を行い、問題の早期発見・対応につなげている。これらの一連の業務は、開発工程のフローに組み込まれる形で管理されている（図 4.13-2）。

合は、日本の輸出規制に抵触しないように使用 OSS に暗号モジュール等が含まれないか確認している。

¹¹² システムが業務で使用されるようなクリティカルなものなのか、もしくは生産性を高めるといったレベルの影響度が小さいものなのかといった観点から重要度が判断されている。

¹¹³ 損害保険ジャパン提供資料を基に作成

リスク分類欄の「✕」印は対策を打つ必要性があることを示している。

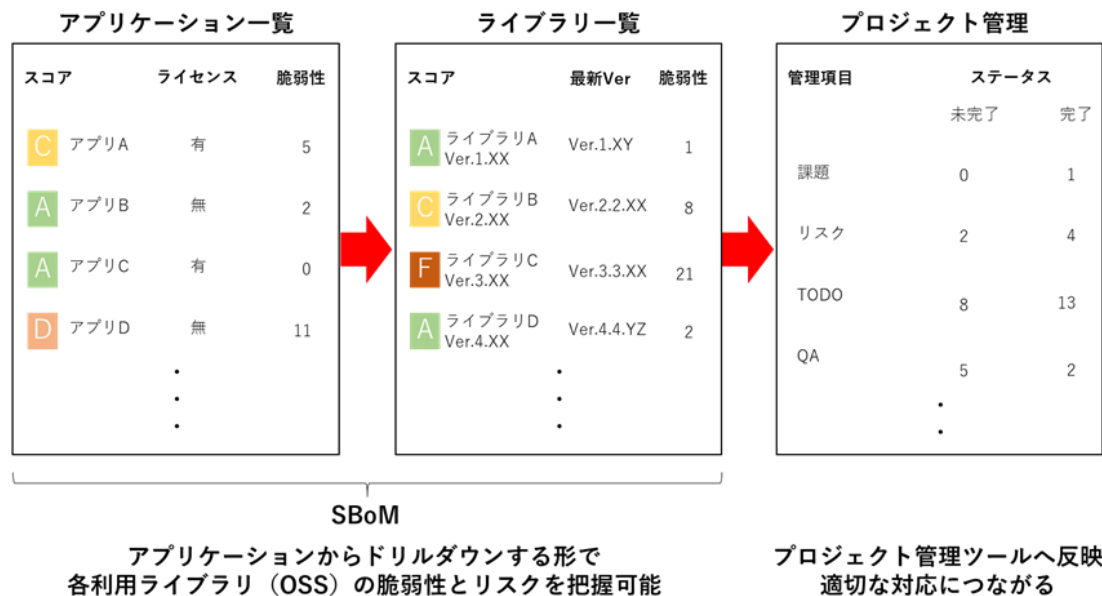


図 4.13-2 開発中、ツールにて作成される SBOM の利活用イメージ¹¹⁴

③最終納品及び運用

システムの納品前に、認識外の OSS がないか、システム全体をツールで再スキャンし、最終確認を行う。同時に、システムを稼働させた状態で解析（動的解析）を行うことで、②の静的解析で確認できなかったリスクの検出も行う。最終確認の結果を反映した SBOM は、システムごとにデータベースに蓄積され、日々の脆弱性管理に活用される。具体的には、当該ツールが日々取得する外部からの脆弱性等に関する情報を基に、その脆弱性等の影響を受ける可能性があるシステムの担当者に対して自動でアラートが発出される。

また、必要に応じて、損害保険ジャパンの OSS 対応組織が、アラートが発出されたシステム担当者に対してフォローを行う等、脆弱性対応が適切に完遂されるような仕組みづくりがなされている。

なお、以前は使用 OSS を手作業で管理しており、データベース上に登録される OSS の名称が統一されていないために使用状況の確認が煩雑であった。しかし、②③における商用ツールを用いたデータベース作成の自動化を実施してからは、そういった運用面の問題も解決された。

¹¹⁴ 損害保険ジャパン提供資料を基に作成

4.14. Visional グループ ～自社状況に対して最適なツールの利用～

■ 企業情報・本事例集における位置づけ

本社	東京都渋谷区
業種	情報・通信業
従業員数（グループ従業員数）	1,385 名人（2021 年 1 月末時点現在）
商流	Sier 商流/システムエンドユーザー（IT サービス）
観点	「脆弱性対応」

■ 本事例のエッセンス

- ・ 開発現場で OSS 利用が比較的自由に行える中で、一度に多数の脆弱性が検知される自社の状況に合ったツールを開発し、事業化した。
- ・ ツールの自社開発にあたっては、社内外におけるテストを繰り返してフィードバックを得ることで、その機能をブラッシュアップした。
- ・ ツールを利用した OSS 管理をルール化し、セキュリティ部門が中心となった脆弱性対応態勢を整えた。

■ 背景・課題

ビズリーチ¹¹⁵（現・Visional グループ¹¹⁶）では、IT 企業として、開発現場のエンジニアが最適な技術を自由に選択・利用してサービスを生み出せる点を大切な価値観と捉えている。その中で、開発工数の削減等の理由から、OSS は積極的に利用されていた。

一方で、事業が複数軌道に乗っていく中で、各サービスにおける使用 OSS が多岐にわたり、手作業の管理では、多量に発生するソフトウェア脆弱性に対応し切れない状況となっていた。

¹¹⁵ビズリーチは、Visional のグループ会社である。本事例で取り上げる OSS 脆弱性管理ツール「yamory」は、当初、ビズリーチから 2019 年にリリースされている。その後、グループ経営体制への移行に伴い、Visional グループが誕生し、新規事業開発を担うビジョナル・インキュベーションが設立された。現在は、ビジョナル・インキュベーションが yamory の開発・販売を担っている。

¹¹⁶ ビジョナル・インキュベーション、ビズリーチなどを含む企業グループ

■ 取組

<自社状況に対して最適なツールの開発と事業化>

ビズリーチでは、ソフトウェア脆弱性にしっかりと対応を行い、セキュリティを担保しなくてはならないとの課題意識から、最適な OSS 管理手法を模索した。これまでの自由に技術を選択可能な風土を尊重しつつ、最低限遵守すべきセキュリティ基準や管理の仕組みを作る必要があった。

そのような中、OSS 管理に課題を感じていたビズリーチの技術者が、複数の企業に対して OSS 管理に関するヒアリングを実施したところ、他社も同様の課題認識を持っていることが判明し、セキュリティリスクを効率的に管理するための新しいツール開発が必要との認識が芽生えた。

自社内で使用するツールの開発では、社内リソースを確保することが難しいが、他の企業にもニーズがあるのであれば、セキュリティ対策ソフトウェア事業として開発すれば良いのではないかと考え、社内の新規事業立案制度を活用し、OSS 脆弱性管理ツール「yamory（ヤモリー）」の開発がスタートした。

<社内外のフィードバックを基にしたツール機能のブラッシュアップ>

yamory の開発では、プロトタイプ作成と社内テストによるフィードバックを何度も繰り返すことでその性能を高めていった。さらに、製品コンセプトが固まった段階においては、製品化に向けた社外テストも実施した。社内テストの実績を踏まえて提案を行うことで、テストに参画した企業は数十社にのぼり、結果として多くのフィードバック¹¹⁷を得ることができた。

これらのフィードバックを基に、脆弱性を多量に検知した際、その緊急度に応じて脆弱性のトリアージ（緊急度に応じた分類）を支援する機能や、それら脆弱性情報を分かりやすく一覧化して表示するダッシュボードにおいて工夫を凝らすことができた（図 4.14-1）。これらの機能により、OSS 管理や脆弱性対応を格段に効率化することが可能となった。

¹¹⁷ ライセンスのリスクやミドルウェア/OS レイヤの脆弱性の管理機能の実装等の要望を得ている。



図 4.14-1 トリアージ機能によりレベルごとの脆弱性が明示されている yamory のダッシュボード¹¹⁸

<【脆弱性対応(v14)】開発したツールを用いた社内ルールの整備>

現在、Visional グループでは、yamory を使用した OSS 管理をグループ全体でルール化している。サービスの開発チームは、リリースまでに必ず yamory を用いて OSS のスキャンを実施しなくてはならない。さらに、開発及びソースコードの管理をしている GitHub に連携し、使用している OSS を日々自動でスキャンする。スキャンにより作成された OSS リストは、日々収集される OSS の脆弱性情報と組み合わせること、リリース後も自社サービスにおける脆弱性有無の検知に利用される。

脆弱性情報は、yamory のダッシュボード上でリスクレベルに応じて分類（トリアージ）され、サービスごと、もしくは全社俯瞰的に表示される（図 4.14-2）。これを各事業部、及びグループ全体のセキュリティ部門が定常的に確認しており、脆弱性が検知された場合、セキュリティ部門が中心となって各サービスの事業部に必要な対応を指示している。また、半年に 1 度、同じくセキュリティ部門が中心となり、各サービスのセキュリティチェック¹¹⁹をグループ全体で実施している。その際に、yamory を用いて、OSS に係る脆弱性が全て解決されている旨のレポートを作成し、提出しなくてはならない。

¹¹⁸ Visional グループ提供資料より抜粋

¹¹⁹ OSS に関する脆弱性に関するチェックのみならず、製品セキュリティチェックをトータルに行っている。ツールで管理しきれない OS、主要なミドルウェア、ソフトウェアの EOL 等のチェックもこの際に行う。

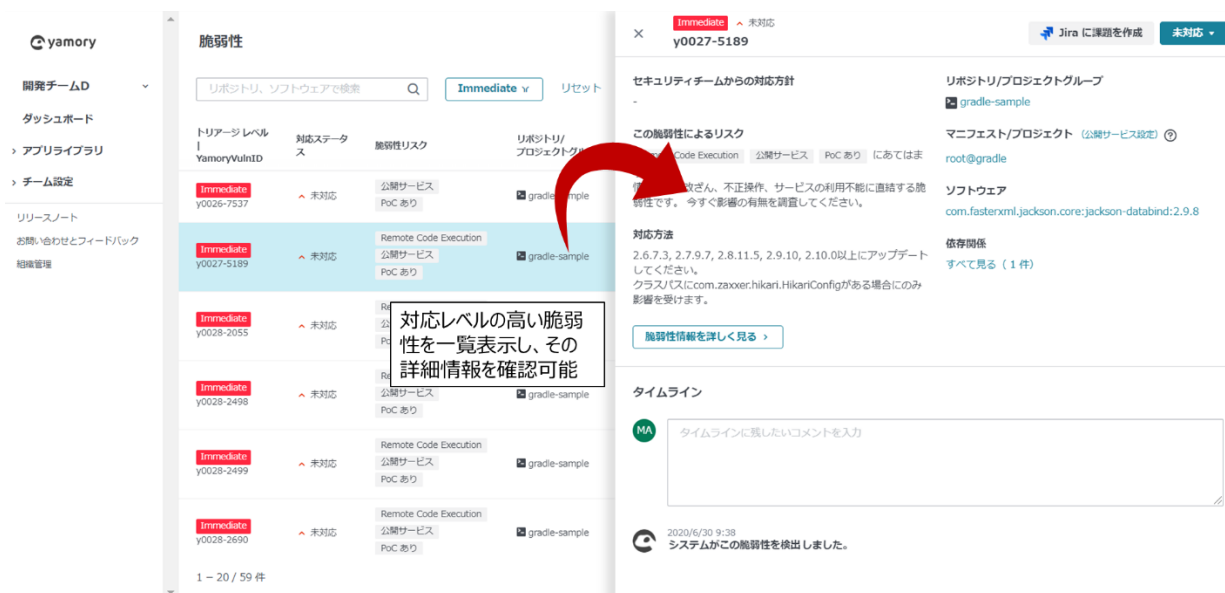


図 4.14-2 対応方針等を含め詳細を確認することが可能な脆弱性の一覧画面¹²⁰

yamory を使用した OSS 管理をグループ全体でルール化するにあたり、開発チームは、トリアージ機能によって OSS に係る脆弱性対応の負荷軽減につながることを、各サービスの開発現場に実績を示してアピールした。実際に、各開発現場で独自に脆弱性管理を行うことが負荷となっていた事実もあり、ツール利用が受け入れられていった。最終的に、セキュリティ部門も巻き込むことで OSS の適切な管理を効率的に行わなければならないという意識が高まり、グループ全体でのルール化に至った。

<今後の展望>

今後は、まだ yamory でカバーできていない EOL の管理、ミドルウェアや OS の脆弱性管理についても機能を追加し、より集約的に OSS を管理できるよう研究開発を進める予定である。

なお、yamory の営業現場には、顧客の OSS 管理に対する様々な声（機能追加の要望や運用に関する相談等）が寄せられている。そこから、OSS 管理における各社の対応は、その企業が扱う情報の質や業種等によって様々であり、OSS 管理に対応し切れていない企業も一定数ある状況が分かっている。ビジョナル・インキュベーションでは、引き続き、製品（ツール）、及び自グループ内の運用手法のブラッシュアップを通じて顧客提供価値の向上につなげたいと考えている。

¹²⁰ Visional グループ提供資料を基に作成

4.15. サイボウズ ～OSS エコシステムに貢献する OSS ポリシー～

■ 企業情報・本事例集における位置づけ

本社	東京都中央区
業種	情報・通信業
従業員数（連結）	741 人（2019 年 12 月末現在）
商流	SIer 商流/システムエンドユーザー（IT サービス）
観点	「ライセンス」「個の能力・教育」「コミュニティ活動」「選定評価」「脆弱性対応」「組織体制」

■ 本事例のエッセンス

- OSS のエコシステムに貢献するため、部門横断的な組織を立ち上げ、全社的な OSS ポリシーを策定した。
- PSIRT が中心となり、ソフトウェア脆弱性情報のハンドリングや脆弱性報奨金制度の運用等を行い、自社サービスのソフトウェア脆弱性撲滅に向けた取組を徹底している。

■ 背景・課題

サイボウズでは、主力の Web サービスである kintone や cybozu.com のインフラ等において、多量の OSS を使用している。一方、数年前までは、OSS を利用する際には、個別に開発者及び法務がライセンスを確認し上長承認のもと、ライセンスを遵守するといった形で利用しており、OSS の開発を含む全社的な方針はなかった。

■ 取組

<【ライセンス(L15)】【個の能力・教育(P15)】【コミュニティ活動(C15)】OSS のエコシステムに貢献する OSS ポリシーの策定>

サイボウズでは、OSS の発展に寄与する存在でありたいとの思いから、社員のオープンソース活動を支援し、貢献を促すような OSS ポリシーの策定を目的として 2017 年に OSS 準備室を設立した。OSS 準備室は、OSS に関する全社各部門の代表者からなる部門横断型の組織である。

OSS 準備室は、主に OSS ポリシーの策定及びその全社ルール化と OSS ポリシーを補完する細則である OSS ガイドラインの整備を行った。OSS ポリシーは、著作権、特許、商標のライセンス等の取り扱いについて定めている。なお、サイボウズの OSS ポリシーはパブリックドメイン（CC0）で公開されている（表 4.15-1）。

表 4.15-1 サイボウズの OSS ポリシーの章構成¹²¹

0. 本文書について	-
1. 前文	-
2. 著作権	2.1. 著作権の帰属 2.2. 著作権の譲渡
3. 従業員個人の著作物のオープンソース化および当社商標の使用	3.1. Disclaimer（免責条項）の記載 3.2. 当社商標の使用
4. 当社著作物のオープンソース化	4.1. OSS ライセンスでの公開 4.2. 他者 OSS への提供
5. 他者 OSS の利用	5.1. 他者 OSS のライセンスの確認 5.2. ライセンス文書等の取扱い 5.3. 他者 OSS の不具合に関する報告
6. 当社 OSS への他者著作物の取り込み	6.1. 他者著作物の提供があった場合 6.2. コントリビューターリストの管理
7. ライセンス違反への対応	7.1. 当社 OSS のライセンスに違反する利用への対応 7.2. 当社内での OSS ライセンス違反への対応

サイボウズでは、自社開発したソフトウェアを OSS 化する機会も多くあるため、OSS ポリシーではその際に遵守すべき事項も定めている。その中には、簡略な承認ステップで OSS 化を可能とする規定もあり、OSS エコシステムへの貢献の姿勢が反映されている。さらに、OSS の利活用促進に向けて開発者のモチベーションを高めるため、社員が自発的に作成したソフトウェアは、サイボウズのライセンスや極秘情報を含まない限り、開発した社員の著作物とする規定もある。以上のように、OSS ポリシーは、OSS の幅広い利活用を想定した内容となっている。

こういった OSS 利活用に対する積極的な姿勢は、OSS 化したソフトウェアのバグに関するフィードバックを外部からも得られるといった実務的なメリットの享受や、採用活動における技術力や自由な社内風土のアピール材料としての活用等、さまざまな効果を生んでいる。

サイボウズは、kintone 等のグループウェアを主力サービスとする会社であり、グループウェアを通じて全社員が容易に意思疎通できる環境が備わっていた。その中で、OSS 準備室の立ち上げや、ポリシーの策定・公開においては、当時の運用本部長が中心となって、トップダウンで必要性を啓発し、取組を強力に

¹²¹ サイボウズの OSS ポリシー

<https://cybozu-oss-policy.readthedocs.io/ja/latest/>

推進した。

なお、OSS 準備室は、新設された OSS 推進室¹²²にその機能を引き継ぎ 2018 年に発展的に解散している。現在は、OSS 推進室が OSS ポリシーと細則を定めた OSS ガイドラインのメンテナンスを行っている。

<【選定評価(S15)】【脆弱性対応(V15)】【組織体制(O15)】PSIRT が軸となった多様なソフトウェア脆弱性対応>

サイボウズでは、サービスにおいて OSS を利用する際には、前述の OSS ポリシー、OSS ガイドラインに従って、開発チームが利用したい OSS の調査を行わなくてはならない。調査の結果、ライセンスが社内で定めているガイドラインに合致すること、定期的にアップデートされていることが確認されれば、その OSS を利用することができる。ライセンスがガイドラインに合致しない場合や定期的なアップデートがされていない場合でも、法務による審査や自社でバグ等の対応ができるかというチェックを経ることで、その OSS は利用できる場合もある。

各開発チームは、それらの調査等を経た後に、使用 OSS の情報をデータベースに登録する。その情報は、最終的に開発チーム内のコードレビューを経て、抜け漏れがないことが担保される。サイボウズの PSIRT は、そのデータベースを定期的に参照しており、バージョンアップ等の情報があつた場合に開発チームへフィードバックするサイクルができています。

あわせて、PSIRT は、OSS のレポジトリやサイト等の一次情報や JVN、セキュリティニュースサイト等からソフトウェア脆弱性の情報を日々収集している。ソフトウェア脆弱性情報が得られた際には、関連する OSS を使用しているサービスがあるか、PSIRT がデータベースを検索し、確認をしている。該当するサービスがある場合、PSIRT が中心となり情報をハンドリングし、そのサービスの責任者にグループウェア (kintone) 上でアラートを出している。

また、社内である OSS を新たに使用する際、その OSS をマスター登録し、そのマスターを起点にデータベース内でサービスとその使用 OSS を一元的に紐づけている。そのため、ソフトウェア脆弱性情報における OSS の名前情報にばらつきがあつた場合でも、社内では迅速に対応することができる。

さらに、サイボウズの PSIRT では、脆弱性報奨金制度を運用している (図 4.15-1)。これは、サイボウズの自社製品の脆弱性を発見、報告した外部の方に報奨金を支払う制度であり、サービス自体の脆弱性や、脆弱性のある OSS 使用の指摘等がその対象となっている。指摘に対しては検証用の環境を提供する等、真摯な対応の下に報奨金支払いの判定を行っている。また、報奨金支給額が高額にならないよう、自助努力として自社でセキュリティ専門企業の脆弱性診断を受ける等、外部からの指摘を極

¹²² 2021 年 1 月現在 7 名が所属している。

力減らす工夫や、外部からの指摘に対して適切に対応できる社員のスキル・リテラシーを養う等、脆弱性を撲滅するため、この報奨金制度を起点とした PDCA サイクルを回している。



- サービスのセキュリティ品質向上に向けて2014年から実施している取り組み
- 2019年度は、489件の着信、193件の認定数（暫定）、1,535万円の報奨金支払金額（暫定）であった
- バグハンターの方を集め、脆弱性を見つけてもらいイベント等も開催

図 4.15-1 脆弱性報奨金制度のホームページ及びその概要¹²³

¹²³ 脆弱性報奨金制度のホームページ

<https://cybozu.co.jp/products/bug-bounty/>

2019 年報奨金制度の結果

<https://blog.cybozu.io/entry/2020/05/07/110331>

4.16. 三菱電機インフォメーションシステムズ（MDIS） ～OSS の特性を踏まえたソリューション展開～

■ 企業情報・本事例集における位置づけ

本社	東京都港区
業種	情報・通信業
従業員数	1,279 人（2021 年 3 月末時点）
商流	製品商流/最終製品メーカー
観点	「選定評価」「保守・品質保証」「コミュニティ活動」「脆弱性対応」「ライセンス」

■ 本事例のエッセンス

- OSS を含んだソリューションを提供するにあたり、OSS を長期間利用する上での脆弱性管理やアップデート対応に係るコストの考え方を顧客と事前に合意している。
- 利用する OSS を適切に把握し、管理するためには、コミュニティとの関わりが重要であり、バグを発見した際はコミュニティに対してバグを報告するだけでなく、改善策も含め連絡することで貢献している。

■ 背景・課題

三菱電機グループでは、開発する製品やソリューションにおいて OSS を活用しており、全社的にも OSS 管理のための組織体制を組み、研究活動も進めている。

MDIS は、OSS を活用した通信事業者、サービスプロバイダー向けのソリューションを提供している。通信業界では、多くの顧客に対して大規模なサービス（携帯電話の SMS/MMS サービス等）を提供するため、有償のソフトウェアを使用すると莫大なライセンス費用を負担することになってしまう。そのため、通信業界は、有償のソフトウェアではなく、無償の OSS の活用に積極的である。

■ 取組

<【選定評価(S16)】オープンソース活用ソリューションの提供>

MDIS では、サービスプロバイダー向けのオープンソース活用ソリューションを提供しており、OSS ベースの

システムライフサイクル全般を支援している。ソリューションの提供先である通信業界では前述のとおり、OSS の活用に積極的であり、提供先からの要求に基づき、MDIS から OSS を活用したシステムやソリューションを提供している。OSS の活用においては、OSS の調査や選定、開発後の運用維持を重要視している。MDIS では、システムやソリューションを 5 年程度使用することを前提に、その間の運用を維持するため、OSS コミュニティに関わりながら、パッチや脆弱性情報を監視し、必要に応じてパッチの適用等を行っている。

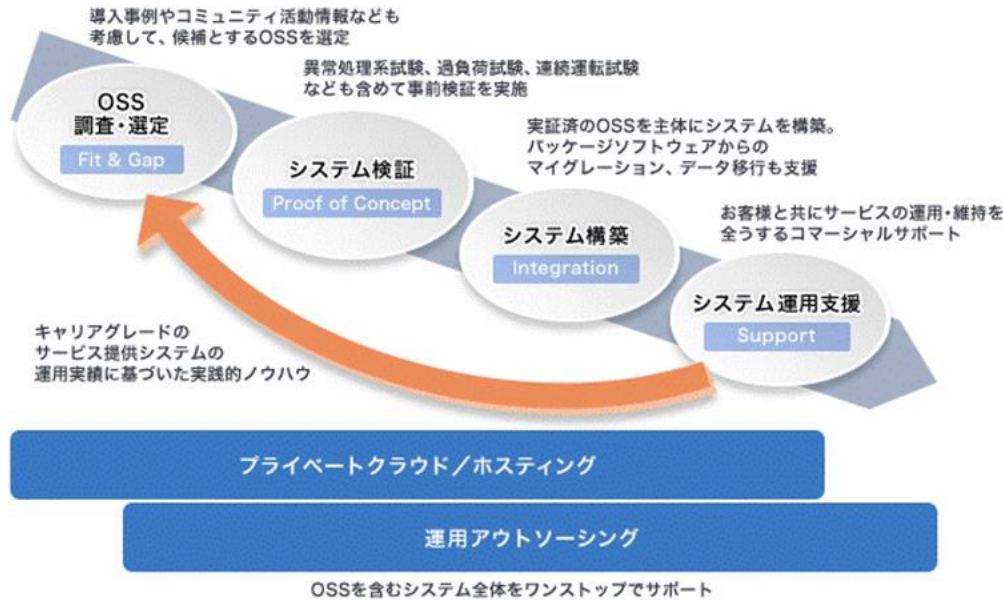


図 4.16-1 オープンソース活用ソリューション（通信事業者、サービスプロバイダー向け）概要イメージ¹²⁴

<【保守・品質保証(M16)】【コミュニティ活動(C16-1)】オープンソース活用ソリューションの提供>

OSS を 5 年という長期間使用するにあたっては、OSS コミュニティが開発や脆弱性対応を含むサポートを終了するというリスクがある。そのため、開発の初期段階で推測されるリスクを洗い出し、発生し得る事態と対策する場合の保守コストについて、顧客に事前に説明し合意している。特に、開発・サポートが終了した後も対策コストが発生するリスクがあることを顧客も理解して運用していくことが重要である。

OSS を改修した際、開発コミュニティへどのように還元するかに苦労した経験もある。OSS のコミュニティや Github は個人開発者を前提として仕組みを作っているため、企業が組織的に改修した場合でも、そのまま組織の成果物として扱うことが難しい。そのため、改修に関わる設計～プログラミング～試験に関与した社員や協力会社の間で貢献度を算定し、また契約にも照らし合わせた上で、メンバー間で納得感がある還元方法を決定する必要があった。また、以降のサポート体制・分担も想定し、連絡先の表示方法を決定する必要

¹²⁴ 通信事業者、サービスプロバイダー向けオープンソース活用ソリューション紹介ページより抜粋
<https://www.mdiss.co.jp/service/oss/>

があった。

<【脆弱性対応(v16)】脆弱性情報の管理>

構成管理ツールに付属している脆弱性情報のプッシュ機能も参考に、OSS の脆弱性を公開している様々なサイトで情報収集している。また、OSS コミュニティから脆弱性を含む情報を発信するメーリングリストがあり、MDIS の開発メンバーのメーリングリストで加入している。他方、コミュニティ活動が無く、OSS コミュニティから脆弱性情報を把握できないような場合は、脆弱性管理ツールを利用して、使用する OSS の脆弱性解析を行うこともある。

<【ライセンス(L16)】構成管理の方法>

OSS の利用が多くない、自らがソフトウェアを選定している、及び事業部門内で開発が完結しているという部門では、SBOM を積極的には利用していない。ただし、外部委託した場合は、知らずに OSS が組み込まれている場合もあるため、構成管理ツールでソフトウェアを解析し、OSS が組み込まれているかを確認している。ライセンス違反を防ぐことがコンプライアンス面、ビジネス面でも重要と考え、構成管理ツールを使用したソフトウェア解析を、ライセンス違反の自己防衛策として取り組んでいる。

<【コミュニティ活動(c16-2)】OSS コミュニティへの関与>

OSS コミュニティは、SIer の立場で参加し、主に OSS の選定評価や運用維持のために情報を収集する場として活用している。OSS コミュニティへの貢献は、OSS のバグを発見した際、単なる指摘だけではなく改善策も含めて連絡し、コミュニティと連携するようにしている。

MDIS が提供するソリューションは、“継続的なコマーシャルサポート”を特長としており、OSS 自体のサポート、もしくは商用レベルにパッケージして商用ディストリビューションとして展開することが、MDIS の役割と考えている。役割を担うには、OSS に関する情報を十分に獲得することが必要である。そのためにも、バグや脆弱性情報を発見した際は、OSS コミュニティと連携し改善するなど積極的に関わるようにしている。

4.17. OSSTech ～OSS を活用したソフトウェア管理～

■ 企業情報・本事例集における位置づけ

本社	東京都品川区
業種	情報・通信業
従業員数（連結）	19 人（2021 年 8 月時点）
商流	SIer 商流/システムエンドユーザー（IT サービス）
観点	「ライセンス」「脆弱性対応」「サプライチェーン管理」

■ 本事例のエッセンス

- OSS を利用したソリューションでは、提供先であるユーザーとの OSS の取り扱い方法や契約形態で留意する点が存在する。
- 構成管理では、ビルドシステム等の OSS に備わっている機能を活用しながら、ソフトウェアの依存関係を管理している。

■ 背景・課題

OSSTech では、認証系の OSS である OpenAM¹²⁵を利用したソリューションを提供しており、セキュリティに配慮した利用や運用が求められる。また、専門的な OSS 利活用を行いつつ、少人数でも管理できるような手法が求められている。この手法のひとつとして、開発環境の機能を活用した管理に取り組んでいる。

■ 取組

<【ライセンス(L17)】取引企業と契約に関する取り決め>

一般的な受託開発によるソフトウェアと比較して、OSS を利用する際にはユーザーに提供するコンポーネントのライセンスや、関連して開発した著作物の扱い（サブモジュールや開発した部分のドキュメント等）を含む契約などの点に注意が必要である。

OSSTech ではユーザー向けに開発する機能やモジュールのライセンス形態を、GPL や Apache License

¹²⁵ シングルサインオン環境を提供する OSS

など、コアコンポーネントに基づくライセンス形態に合わせているが、自社の優位性として利用条件を限定したいユーザー側の思惑とズレが生じることがあるため、契約の段階で OSS のライセンスを適用する範囲を明確にして合意しておくことが重要である。

<【脆弱性対応(v17)】脆弱性情報の管理>

OSSTech で利用及び提供している OpenAM など最近の OSS は、GitHub を利用することで、コミュニティが共同で開発、管理している。また、GitHub のサービスで、リポジトリに登録されている OSS の脆弱性情報が送られてくるため、それらの OSS を利用している開発者は送られてきた情報を確認し、必要に応じて対処している。このようにサービスや仕組みを利用してソフトウェアの安全性を担保できるようになっている。

脆弱性管理において確認・対応すべき箇所は OSS や製品によって異なるため、各製品の開発担当者が OSS のアップストリームからの情報を収集している。開発担当者が情報収集することで効率的で的確に脆弱性を収集し、管理することが可能である。また、アップストリームにおける脆弱性情報を、開発者同士のメーリングリストや BTS（バグトラッキングシステム）¹²⁶ で確認することで対処漏れを防止している。

<【サプライチェーン管理(s17)】OSS を利用したソフトウェアの管理>

OSSTech では、OSS を利用したソフトウェアの管理について、ソフトウェアのユーザーへ SBOM を提供する場合には、同時に自動で脆弱性の確認を実行するツールも提供し、脆弱性のアップデートが実施できることが重要であると考えている。また、ユーザー自らが実施できない場合は、専門企業のサポート体制を含めたアウトソースが現実的であると考えている。その理由は下記のとおりである。

- 多くのユーザーは、OSS の脆弱性アップデートを含めたすべての対応をサポート費でベンダ側に任せたいと考える。ユーザーは、セキュリティを担保することが目的であり、ソフトウェアのコンポーネントを管理することの意識は高くない。
- 組み込み製品の場合は、SBOM を Excel で管理することは可能と考えるが、依存ライブラリが 100 や 1,000 を超える場合は、管理負荷が高くなり困難となる。
- 製品のアップデートがなければ SBOM は基本的に変わらない。しかし、例えば、製品が OS 標準ライブラリをダイナミックリンクしている場合、依存ライブラリ側に変更があれば、SBOM の更新が必要になることがある。そのため、ユーザーに SBOM を提供しても SBOM を活用し、適切に管理、脆弱性のアップデート等を行えるか不明である。

¹²⁶ ソフトウェアの開発や保守を進める際、プログラムなどに含まれるバグを管理、共有するシステム

4.18. ヤフー ～社員が正しく OSS を使うための体制整備～

■ 企業情報・本事例集における位置づけ

本社	東京都千代田区
業種	情報・通信業
従業員数（連結）	単体：7,167 人（2021 年 3 月末現在）
商流	インターネットサービス事業者
観点	「選定評価」「ライセンス」「脆弱性対応」「個の能力・教育」「組織体制」 「保守・品質保証」「サプライチェーン管理」「コミュニティ活動」

■ 本事例のエッセンス

- ・ ヤフー全体で OSS の取り扱いに関するガバナンスが取れており、複数の審査を経たうえで OSS の利用が可能となっている。
- ・ 技術規定に紐づくガイドラインの中で、利用してよい OSS の基準を定めている。
- ・ 複数のツールを用いることで、抜け漏れのない OSS 管理、ソフトウェア脆弱性対応を行っている。
- ・ 開発において外注することはないが、外注する場合は、発注要件の中に OSS のリストを明示することで OSS の管理、把握を行っている。
- ・ 人事規定の改定や OSS デベロッパー認定制度等、OSS に貢献している社員が使いやすい制度を整備し、社員のコミュニティへの参画を支援している。

■ 背景・課題

ヤフーでは、より良いサービスを素早く提供するために、サービスやそれらを支えるプラットフォームにおいて、非常に多くの OSS を利用している。そのため、OSS 利用に関するガイドラインを定め、OSS 選定から導入、保守運用に至るまで様々な規定を設け、遵守することで高品質なサービスを提供可能としている。また、コミュニティへの積極的参画を通じて自社事業の発展にも大きく影響を与える状況となっている。

■ 取組

<【選定評価(S18)】開発部門、法務部門の連携による OSS 承認>

OSS の選定についてヤフーでは、ガイドラインに法務確認を行った利用してよい OSS を定めている。法務確認が済んでいないライセンスは、基本的に利用できない。規定の審査を受け、オープンソースとして問題の無いもの等を確認する審査プロセスを踏まえた上で、さらに技術責任者の承認プロセスを得た上でなければ利用ができない運用になっている。利用が許可されていない OSS については、上記の審査を早期に行い、問題ない場合は、利用を承認するとともに、下位規則に新たに利用可能になった OSS を追加する運用である。

<【ライセンス(L18)】【脆弱性対応(V18)】【個の能力・教育(P18)】【組織体制(O18)】OSS 利活用のコンプライアンス体制と運用の効率化>

ヤフーが展開する Web サービスでは、OSS なくしてはそのサービスが成り立たない程に多量の OSS が利用されている。その中で、2009 年頃、変化の激しいインターネット業界において、社内での OSS 利用がそれまでも増して急増したことから、社員の適切な利活用を強化する目的で仕組みづくりが開始された。また、世の中のコンプライアンスへの意識の高まりにも配慮し、著作権に関する全社員向けコンプライアンス研修の中で、写真や音楽の他、ソフトウェアについても教育や試験を行っている。ヤフーでは OSS に関する基本的な方針として

- ・オープンソースの利用に関する規定
- ・OSS に対する貢献
- ・ヤフーの中からオープンソースを公開

の 3 点を置き、これらを遵守し開発を進めていくことで高品質なサービス提供を行っている。

現在、ヤフーが開発するサービスにおいて、OSS を利用する際には、ガイドラインで定められたプロセスに沿って利用 OSS を申告した上で、一定のチェックを経なければ利用許可が下りない形となっている。具体的には、ライセンスに規定された権利や義務を遵守できるか、また将来に渡って安定的に利用できるか等、OSS 利用者一人一人がコンプライアンス的な観点を含む様々なチェックが出来るよう教育を実施している。

OSS の具体的な選定について、前提となる OSS ライセンスの利用可否は、条文の内容を法務部で確認し、利用してよい OSS ライセンスの場合、確認済みライセンスとしてリスト化して社内公開するようガイドラインに定めている。OSS 利用者は、利用したい OSS のライセンスがリストの中に含まれているかを OSS の利用前に確認する。法務部の確認が済んでいないライセンスは基本的に利用できない。

また、ライセンス以外にも著作権や知的財産権などの様々な面で利用に問題が無い OSS かどうかを確認する選定コストを下げるため、全社的に利用する頻度も高く、そうした選定コストも高い OSS については、専門の部隊が事前に審査し、選定作業をせずとも利用可能な OSS として、ガイドラインの下位規則に新たに追加していく運用である。

それ以外の OSS については、確認済みライセンスか確認し、利用しても問題ないかの選定作業を終えた

上で、OSS 利用を申請し、その内容に問題がないかを技術責任者が確認し、承認するプロセスとなっている。なお、選定時のライセンス、その他のチェックには、様々なツールを組み合わせて利用している。

また、ソフトウェア脆弱性対応の取組として、CSIRT を中心とした対応とツールを利用した対応の 2 つがあげられる。リスクの高い脆弱性の場合、CSIRT が中心となり規定に基づく必要な対応を各事業部に要請している。一方、それ以外の脆弱性の場合、ツールによって脆弱性情報が自動で検知され開発者自身にアラートが出る仕組みとなっており、ソフトウェア脆弱性に迅速に対応可能となる。ヤフーでは、抜け漏れを防ぐため、開発から運用までの各フェーズで複数のツールを使用して脆弱性チェックを行っている。

さらに、ヤフーでは、自社または他社とのコラボレーションにより開発したソフトウェアを OSS として公開する機会がある。そういったコミュニティへ貢献をする場合、ライセンス違反や第三者の権利を侵害することがないように、技術責任者によるチェックを経なくてはならない仕組みも構築している（図 4.18-1）。

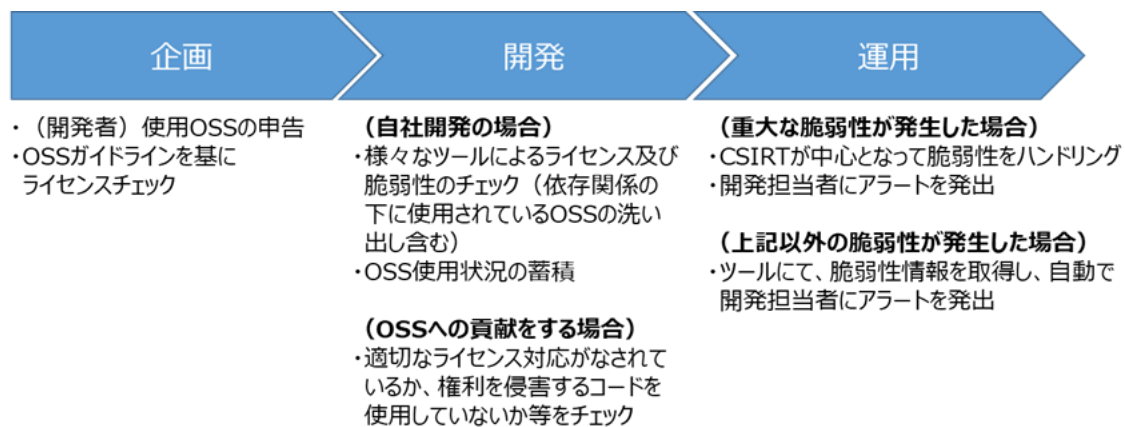


図 4.18-1 ヤフーにおける OSS 管理の全体像¹²⁷

これらの運用を軌道に乗せるため、CTO¹²⁸からトップダウンで各事業部門を統制し、開発者向けの研修プログラム等を実施していった。コンプライアンスやセキュリティへの意識を浸透させるため、開発者や事業推進担当者に対して説明会等を開催し、その中で対応の必要性を論理的に訴え、丁寧にコミュニケーションを取ることで理解を得た。

¹²⁷ ヒアリングを基に作成

¹²⁸ Chief Technology Officer: 最高技術責任者

<【保守・品質保証(M18)】コミュニティが定義したライフタイムサイクルの準拠>

OSS を利用したサービスの使用年数については、OSS コミュニティが定めるライフサイクルに沿って使用するよう運用方法を示したリファレンスを提供しており、基本的にはそのライフサイクルに準拠する前提で利用している。しかし、切り替えができない場合など EOL を過ぎても自社の責任の下、利用を続けるケースはある。EOL を過ぎても継続利用するものについては、他社の有償サポートの利用も検討することとしている。

<【サプライチェーン管理(S18)】社外への開発委託>

ヤフーは自社開発が多く、実際に社外へ開発委託するケースは基本的にはないが、仮に外注する場合は、発注要件の中に OSS のリストを明示することを記載し、これを受け入れ条件にすることとしている。リストに記載する OSS の粒度については、明示的に利用しているライブラリ、フレームワークベースとし、ツールを利用することによって詳細の OSS を把握する想定である。

<【コミュニティ活動(C18)】OSS に貢献している社員が使いやすい制度>

ヤフーは、前述した自社開発ソフトウェアの OSS 化や、他社とのコラボレーションによる OSS 開発、もしくはコミュニティへの積極的参画を通じて OSS のエコシステム発展に貢献している。これらの活動は、OSS の改善による自社事業への還元のみならず、開発者の技術力向上にも効果をあげている。

コミュニティ活動に係る特筆すべき取組として、OSS への貢献を社員が自由に行えるようにするため、会社が認めた場合に、職務上作成した成果物であっても個人に帰属させることができるよう、人事規程上で手当を行っている。また、OSS デベロッパー認定制度を運用している。これは、ヤフーが戦略的に利用する OSS への貢献の機会や、自社製の OSS を開発するような社員が増えてきたことから、社内外で活動・活躍しやすい環境を整備するために設けられた制度である。具体的には、ヤフーが戦略的に採用している OSS に対して、コミットとして貢献している社員をデベロッパーとして認定している。また、対象 OSS 開発や関連する活動について一定額の活動予算枠¹²⁹を付与する等、そのコミュニティにおいてコミットとして活動できる環境を提供している。認定は、CTO 直轄のデベロッパー認定制度事務局が、「対象となる OSS の重要度」「コミュニティからのコミットとしての評価」「コミュニティにおけるコミットとしての活動実績」等を総合的に判断した上で行う。2021 年 10 月までに、のべ 28 名¹³⁰のデベロッパーが認定されている。

さらに、The Linux Foundation 及びその下部団体にあたる Cloud Native Computing Foundation (CNCF) 等、いくつかのコミュニティに対してスポンサード（金銭的な支援）も実施している。

¹²⁹ 開発機器の購入費用や、書籍執筆費用、カンファレンスの参加費や、渡航費や宿泊費など幅広く利用できる。

¹³⁰ ヤフーが戦略的に利用している OSS のコミュニティで実績を積み上げて認定される場合もあれば、そのコミュニティにおいて既に実績を積み上げている人材を採用し、認定をする場合もある。

4.19. ラキール ～OSS の選定会議と検証ツールの併用による開発と管理の効率化～

■ 企業情報・本事例集における位置づけ

本社	東京都港区
業種	情報・通信業
従業員数（連結）	単体：402 名 連結：474 名（2021 年 4 月現在）
商流	インターネットサービス事業者
観点	「組織体制」「選定評価」「ライセンス」「脆弱性対応」

■ 本事例のエッセンス

- OSS の利用を全社的に促進しており、新しい OSS を利用する際には、開発部と OSS 管理部門のリーダー同士がリスクの確認や懸念点を議論する会議で利用可否を決定する。
- OSS 管理の抜け漏れを防ぎ、開発者と OSS 管理者の業務を効率化するために検証ツール（ソフトウェア構成分析やライセンスの検証を行うツール。以下、同様。）を導入している。

■ 背景・課題

OSS はコミュニティがライセンスを変更する場合もあり、自らが利用している OSS を正確に把握し、最新の OSS ライセンス情報を収集しないと、知らぬうちにライセンスに違反してしまう可能性がある。この解決方法として検証ツールを導入し、OSS 管理の確実性の向上や効率化に取り組んでいる。

■ 取組

<【組織体制(019)】OSS を管理する組織体制>

OSS に関係する部門として、OSS を自社サービスに利用するプロダクト開発本部がある。プロダクト開発本部には約 100 名が在籍し、各プロダクトの開発を行うとともに、開発プロダクト毎に管理者（兼任）を置いている。また、本部内の開発者向けのツール管理やプロダクト管理機能の開発を行う部門（プラットフォームグループ）が、OSS 管理を兼任しており、すべての開発プロダクトで使用されている OSS の管理及び棚卸を行っている。

<【選定評価(S19)】新しい OSS の選定会議>

ラキールでは、開発において積極的に OSS を利用していこうという意識があり、開発の効率化のため、開発者が新しい OSS を利用することを止めるのではなく、新しい OSS の利用を促進し、かつライセンスを守り、安全に管理できる方法を目指した。

利用する OSS を選定する際は、OSS を利用するプロダクト開発本部のメンバや関係者及びプラットフォームグループのメンバによる会議において OSS の利用可否を決定することとしており、OSS を利用する企画が上がってきた時に、各プロダクト開発本部とプラットフォームグループのリーダー同士でリスクの確認や懸念点を議論する。会議では、OSS のライセンス、シェア、コミュニティの活性度など各種の情報が共有され、特に、コミュニティの開発がストップしてしまった場合の対応策を検討、確認する。例えば、OSS に脆弱性が見つかった場合でも対処されない場合の対応等である。これまで、選定した OSS のコミュニティの開発がストップしてしまったケースはないが、リスクとして認識しているため、万が一の対応を含め、OSS の利用可否の判断における重要な指標として、コミュニティの活性度等を確認している。なお、上記の運用をルール化して実施している現状に留まらず、今後は OSS の利用や選定に関する体系立てたガイドラインを作成し、明文化による更なる効率化の計画が進行している。

<【ライセンス(L19)】検証ツールの導入による効率化>

ある OSS のライセンス変更を知ったことが、OSS 管理の意識向上のきっかけになった。そのライセンス変更は、商用利用するにはソースコードの公開義務が発生するという内容であった。当該 OSS は、事前に利用用途などを把握できていたため、結果的に影響がなかった。一方で、今後も OSS 利用が増えていく傾向がある中で、他の OSS でも同様に知らないうちにライセンス変更され、ライセンスに違反してしまうケースがあるのではないかと懸念が大きくなっていった。このような状況を未然に防ぐため、“確実な OSS の管理”と“OSS ライセンスの最新情報の収集”が急務であった。

そこで、OSS の管理の徹底を図ろうとしたが、大量の OSS を利用しているため、Excel による管理では抜け漏れが発生し、結果としてライセンス違反に気付けないリスクも感じていた。そのため、検証ツールを導入することとし、その選定・導入に当たっては、現状の管理における抜け漏れの検出可否を評価項目とした。また、検証ツールを導入することで、再帰的に使用されている OSS を検出できるようになった。あるプロダクトでは、利用している OSS を 20～30 程度と想定していたが、検証ツールを利用して検出した OSS は再帰的なものも含めると数百あった。

プラットフォームグループが OSS の管理を一手に担っている体制となっており、従来は開発側が新たに OSS を使用するたびに管理工程が増えていく状態だった。しかし、検証ツールを利用することによって OSS 検知処理および情報整理が自動化されたため格段に工数が軽減された。OSS 管理者の体感的には

1/10 程度になったと感じている。

<【脆弱性対応(v19)】DevSecOps¹³¹による脆弱性対応>

プロダクト開発では、ソースコード管理に GitLab を利用しており、CI/CD¹³²と検証ツールを連携させることで、使用している OSS の検出と、検出した OSS の脆弱性有無のチェックを自動化している。開発サイクルに OSS の検出と脆弱性のチェックを組み込んだことで、OSS 管理及び脆弱性対応の抜け漏れがなくなり、DevSecOps 体制を構築でき、開発スピードが加速した。

¹³¹ 開発(Development)と運用(Operations)が連携する DevOps にセキュリティ対策を組み込むこと。

¹³² Continuous Delivery (継続的デリバリー)、Continuous Deployment (継続的デプロイメント)

4.20. SCSK ～オープンソースソフトウェアへの取組～

■ 企業情報・本事例集における位置づけ

本社	東京都江東区
業種	情報・通信業
従業員数（連結）	14,550 人（2021 年 3 月末現在）
商流	SIer 商流/システムエンドユーザー（IT サービス）
観点	「組織体制」「個の能力・教育」「サプライチェーン管理」「脆弱性対応」「選定評価」

■ 本事例のエッセンス

- OSS の利活用は全社的に重要事項として位置付けられている。開発プロジェクトにおけるリスクの第三者チェック機能である PMO 組織の設立や各事業グループに OSS 管理担当者を設置することで組織体制を構築している。また、e ラーニングや OSS 利活用に関する勉強会を実施している。
- ソフトウェアの部品となる OSS を再帰的に使用することで意図しない OSS が混入していないかを検査するシステム、OSS を調達する際に安全な OSS を選定するシステムを導入し、OSS 利活用に係る課題やリスクに対応している。

■ 背景・課題

OSS を利活用するためには、OSS 利活用に係る課題やリスクを認識した上で、必要な対応を全社的に実施する必要がある。また、ソフトウェア開発において OSS を利用する場合は、利用する OSS が安全かどうか、どのような OSS であるかを把握する必要がある。また、これらの運用を継続していくためにも、体制や仕組みの構築が求められる。

■ 取組

<【組織体制(020)】OSS 管理の組織体制>

開発プロジェクトのリスクを第三者の立場でチェックしプロジェクトの健全な運営を支援する組織として、PMO 組織を設置している。PMO 組織は、各事業グループに設置されている部門 PMO と PMO 施策を統

括・推進する全社 PMO で構成されている。

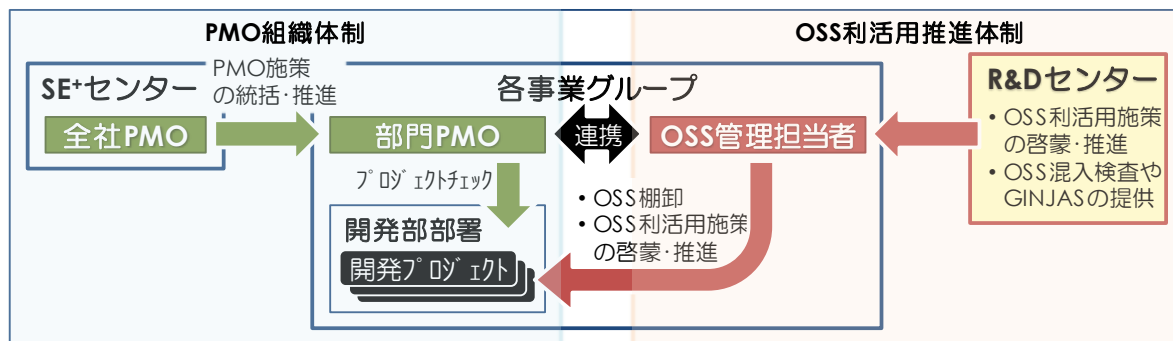


図 4.20-1 PMO 組織と OSS 利活用推進体制の関係¹³³

PMO 組織では、システム開発の計画段階から OSS の利用について問題ないかを確認している。具体的には、お客様の要望等によりシステムで利用が想定される OSS を OSS 選定調達支援システム（Ginjas）に登録し、OSS 一覧を PMO に提出することを求めている。さらに、Ginjas にそのシステムで利用する予定の OSS を登録する際には、ライセンスに記載されている OSS の利用条件をプロジェクトで確認するようにしている。

各事業グループに OSS 管理担当者を設けて、OSS 管理の活動を推進している。具体的には、担当している部署を対象に、四半期ごとに利用している OSS の棚卸を実施している。棚卸のタイミングで、OSS 混入検査の実施確認（実施予定時期、予定通り実施したかどうか、実施していない場合はその理由）、Ginjas 利用の周知を実施している。

システム開発する技術者でも、自らが使用している OSS で求められる対処方法（例えば、GPL ライセンスを利用した頒布の対処等）がわからない場合もあるため、専用の問い合わせ窓口を設けて問い合わせ対応を実施している。問い合わせ窓口は、施策の主管部署である R&D センターに、OSS に熟知した技術者とライセンス情報を確認するメンバーを配置している。問い合わせ窓口の担当者の育成は、e ラーニングや OSS 活用ガイドラインにより OSS ライセンス基礎知識を習得し、他社との勉強会に参加する等の活動も通して、OSS ライセンスの理解を深めている。問い合わせ窓口には、ライセンスの使い方、自社製品に OSS を活用してリリースできるか、お客様と共同で開発した製品の取り扱いなどに関する問い合わせがある。

¹³³ SCSK 提供資料より抜粋

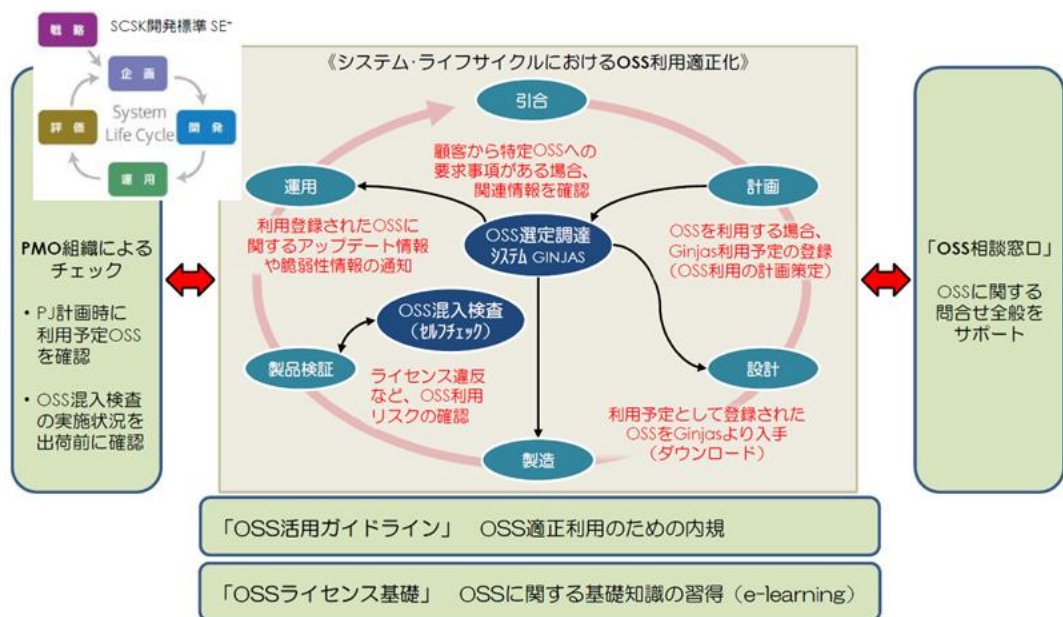


図 4.20-2 OSS 適正活用施策の全体像¹³⁴

<【個の能力・教育(p20)】OSS 管理の方針や知識の社内外への取組>

当社の「情報セキュリティガイドライン」で OSS を含むソフトウェア・ライセンスの適正な利用を定めると共に、開発組織向けに OSS 活用ガイドラインを作成し、グループ全体の現場レベルまで周知している。

また、社内研修の e ラーニングに、OSS ライセンスの解説も掲載し、事務スタッフ以外の社員は全員受講している。なお、研修内容は、問い合わせ窓口で相談のあった中でも重要度が高いものや最新の問合せ内容、最新の相談内容を取り入れ、更新している。

社外向けの OSS に関する啓発活動としては、OSS X Users Meeting¹³⁵という勉強会を開催している。この勉強会は、OSS 利用者や開発者を対象にし、OSS の最新動向や使い方の説明など、情報共有している。

<【サプライチェーン管理(s20)】委託企業への取組>

委託企業に SCSK の OSS 管理に関する取組や仕組みを説明する。委託企業への発注には、SCSK の OSS ガイドラインの確認、受入検証、検証後のフォローの周知徹底や理解が条件になる。

また、社内向け e ラーニングの OSS に関する内容を共有している。さらに、具体的なフォーマットは決めているが、使用している OSS の開示を委託企業へ求めている。

¹³⁴ SCSK 提供資料より抜粋

¹³⁵ <https://technomado.jp/tech/5417/>

<【脆弱性対応(v20-1)】OSS 混入検査>

SCSK は、開発ライフサイクルの製品検証で、OSS の混入検査を実施する。製品開発を外注した場合、受入検証で混入検査を全件チェックしている。OSS 混入検査システムに、コード検査ツール¹³⁶を組み込んでいるが、現場でのセルフチェックを行う上で以下の問題があった。

1 点目は、コード検査ツールは混入の可能性がある OSS 候補を出力するが、候補の OSS 数が膨大な場合、コード検査ツールの出力結果の確認・検証負荷が高い。

2 点目は、出力されるレポートには多くの項目があり、取り扱いに留意が必要となるような OSS を現場レベルで特定することが困難で、作業負荷が高い点である。

これらの問題点を解決するため、SCSK のグループ企業であるベリサーブに依頼し、コード検査ツールをラッピングし、簡単な UI にした。具体的には、様々なオプションが表示される検査ツールの操作画面は使用せず、当社の検査に必要な項目に絞ってプロジェクト名、検査実施者のメールアドレス（検査完了通知用）、ソースコード（ZIP ファイル）登録の 3 ステップで検査を開始できる画面を用意した。さらに、検査結果についても検査ツールの画面で確認するのではなく、Excel のレポート形式で確認できるようにした。Excel のレポートは、利用 OSS ライセンスの利用条件に応じて色分けした検査結果のサマリー、ファイル単位のライセンス情報を記載した一覧シート、確認結果報告用のシート（確認結果報告書）の 3 シートで構成されている。

また、Excel でレポートを出力するセルフチェックシステムへ改修した。さらに、このシステム改修で、検査終了後に通知する機能、検査対象に含まれている可能性の高い OSS を自動判定する機能、対応の優先順位が高いライセンスをもつ OSS を提示する機能を備えることで、開発現場によるセルフチェックが行えるようになった。検査対象に含まれる OSS の判定では、検査ツールが出力する OSS 候補で、一致率が最も高い OSS を選択している。一致率が同じ OSS が複数ある場合は、その中で最も条件が厳しいライセンスの OSS を選択している。対応の優先順位は、利用条件が厳しい順に付けている。コピーレフト型（GPL、AGPL 等）が最優先で、準コピーレフト型（EPL、MPL 等）、非コピーレフト型（Apache、MIT、BSD 等）の順となる。

SCSK での混入検査の精度については、OSS から再帰的に利用される部品についても、納品物に含まれるすべてのファイルを検査することで検出している。なお、ツールには限界があると考えており、ツールでの検査後に、開発者で最終確認を実施するようにしている。具体的には、検査者には、検査結果レポートの内容を鵜呑みにするのではなく、判定結果を確認した上で、OSS ライセンスへの対応方法を確認結果報告書に記載して提出するよう求めている。その確認結果報告書を施策主管部署（R&D センター）への提出をもって最終確認としている。

¹³⁶ OSS 情報データベースとの照合により、検査対象のコードに対して OSS コードを検出する機能がある。

OSS混入検査：開発部署自身によるセルフチェック

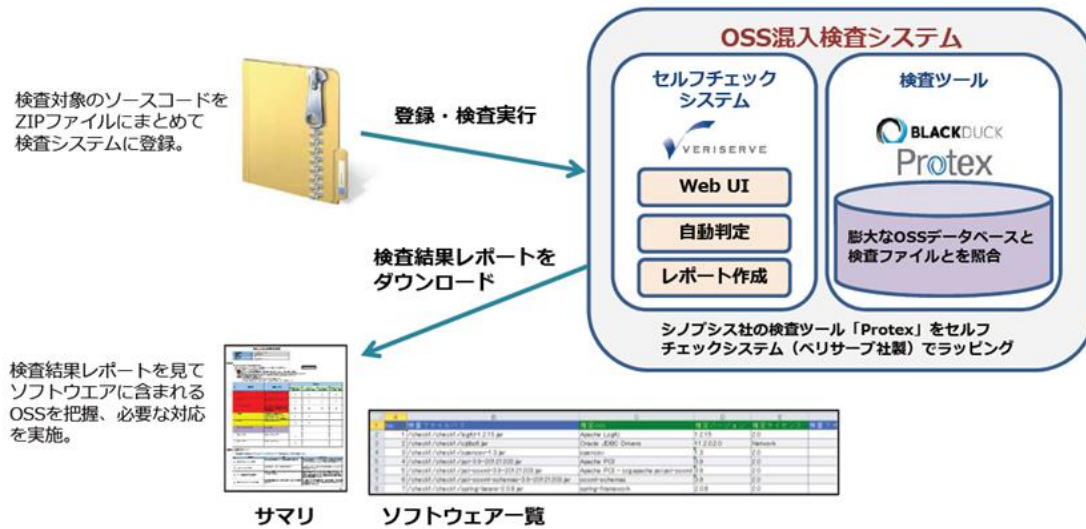


図 4.20-3 OSS 混入検査システムのイメージ¹³⁷

OSS 混入検査システム | 検査登録 | 検査結果 | ログオフ

ログインユーザ
ocu2016-0001

検査状況
登録中

検査登録一覧

<<	<	1	>	>>
OSC-20160322-04 営業支援 修正差分-0001				
OSC-20160322-03 営業支援 クライアントのみ				
OSC-20160322-02 営業支援システム 1回目				
OSC-20160322-01 テスト				

検査登録名

メールアドレス
検査完了時にレポートリンクをメールでお知らせします

解析対象ファイル 参照...

ファイルは登録時にアップロードされます

登録

図 4.20-4 OSS 混入検査システムの画面（解析対象ファイルの登録）¹³⁸

¹³⁷ SCSK 提供資料より抜粋

¹³⁸ SCSK 提供資料より抜粋

OSS混入検査システム

検査登録 検査結果 ログオフ

ログインユーザ
ocu2016-0001

検査状況
検査完了

管理No.OSC-20160322-05 検査登録名：営業支援システム 2回目

No	フェーズ	開始	終了	時間
1	ソースコード登録	開始		2016-03-22 10:49:34
2		終了		2016-03-22 10:49:37
3	ソースコード解析	開始		2016-03-22 10:49:37
4		終了		2016-03-22 10:50:11
5	検査結果レポート作成	開始		2016-03-22 10:50:11
6		終了		2016-03-22 10:55:24

レポートダウンロードURL
<http://xxxxxx/xxxx/report/xxxx/OSC-20160322-05.zip>

検査登録一覧

< 1 >

OSC-20160322-05 ×
営業支援システム 2回目 ×
OSC-20160322-04 ×
営業支援 修正差分-0001 ×
OSC-20160322-03 ×
営業支援 クライアントのみ ×
OSC-20160322-02 ×
営業支援システム 1回目 ×
OSC-20160322-01 ×
テスト ×

図 4.20-5 OSS 混入検査システムの画面（解析結果ステータス）¹³⁹

「OSSライセンス検査報告書」

注意が必要なOSSを色別に表示

No	ファイルパス	検出OSS	検出バージョン	検出ライセンス	検出ファイル
1	/check/check/legit-2.15.jar	Apache Legit	2.15	2.0	
2	/check/check/fojood.jar	Oracle JDBC Drivers	11.2.0.2.0	Network	
3	/check/check/fojood.jar	coarncov	1.3	2.0	
4	/check/check/fojood.jar	Apache POI	3.9	2.0	
5	/check/check/fojood.jar	Apache POI - org.apache.poi.poi-ooxml	3.9	2.0	
6	/check/check/fojood.jar	coarncov	1.3	2.0	
7	/check/check/fojood.jar	springframework	2.0.8	2.0	

注意が必要なOSSを色別に表示

「OSS検査結果ファイル一覧」

No	検出ファイルパス	検出OSS	検出バージョン	検出ライセンス	検出ファイル
1	/check/check/legit-2.15.jar	Apache Legit	2.15	2.0	
2	/check/check/fojood.jar	Oracle JDBC Drivers	11.2.0.2.0	Network	
3	/check/check/fojood.jar	coarncov	1.3	2.0	
4	/check/check/fojood.jar	Apache POI	3.9	2.0	
5	/check/check/fojood.jar	Apache POI - org.apache.poi.poi-ooxml	3.9	2.0	
6	/check/check/fojood.jar	coarncov	1.3	2.0	
7	/check/check/fojood.jar	springframework	2.0.8	2.0	

ファイル毎の判定結果を表示

「OSSライセンス確認結果報告書」

注意が必要なOSSを色別に表示

No	検出ファイル	検出バージョン	検出OSS	検出ライセンス	検出ファイル
1	/check/check/legit-2.15.jar	Apache Legit	2.15	2.0	
2	/check/check/fojood.jar	Oracle JDBC Drivers	11.2.0.2.0	Network	
3	/check/check/fojood.jar	coarncov	1.3	2.0	
4	/check/check/fojood.jar	Apache POI	3.9	2.0	
5	/check/check/fojood.jar	Apache POI - org.apache.poi.poi-ooxml	3.9	2.0	
6	/check/check/fojood.jar	coarncov	1.3	2.0	
7	/check/check/fojood.jar	springframework	2.0.8	2.0	

図 4.20-6 OSS 混入検査システムの画面（解析結果レポート）¹⁴⁰

¹³⁹ SCSK 提供資料より抜粋

¹⁴⁰ SCSK 提供資料より抜粋

<【脆弱性対応(v20-2)】OSS の選定調達支援システム (Ginjas) >

OSS を安全に利活用できるよう Ginjas と呼ぶ OSS の選定調達支援システムを構築している。Ginjas は、製品開発の計画、設計、運用などの各フェーズで OSS の安全な利活用をサポートしている。Ginjas では、事前に安全性が確認できた OSS について、OSS と OSS 自体の SBOM を登録しておく。この OSS を利用する場合は、Ginjas から OSS をダウンロードし、使用する。Ginjas 以外から OSS を入手して使用しないように教育している。

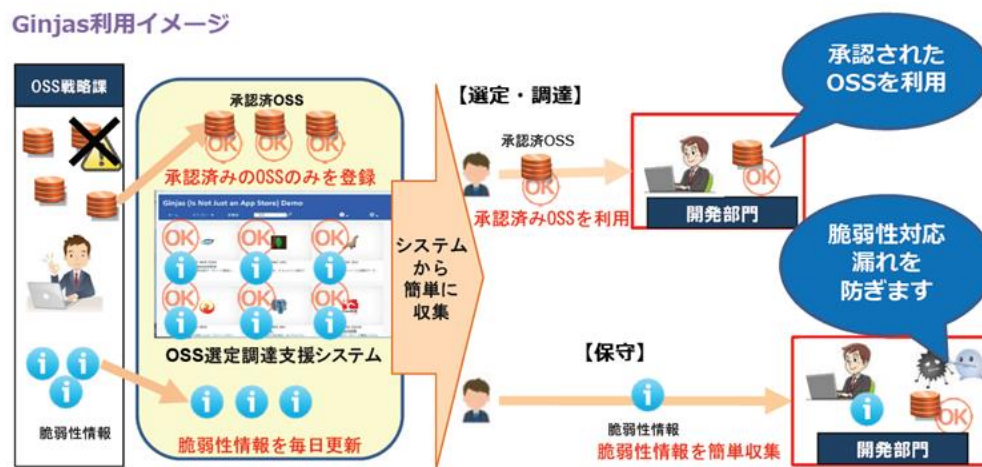


図 4.20-7 Ginjas の利用イメージ¹⁴¹

Ginjas は、脆弱性情報を IPA 等の情報サイトから自動で取り込み、その脆弱性と開発製品を突合して評価結果をレポートとして出力する。この脆弱性情報の取り込みには、脆弱性検知ツールである Vuls を利用し、OSS の脆弱性を確認するようにしている。脆弱性等で問題ある場合は、アラートを通知する。

¹⁴¹ SCSK 提供資料より抜粋

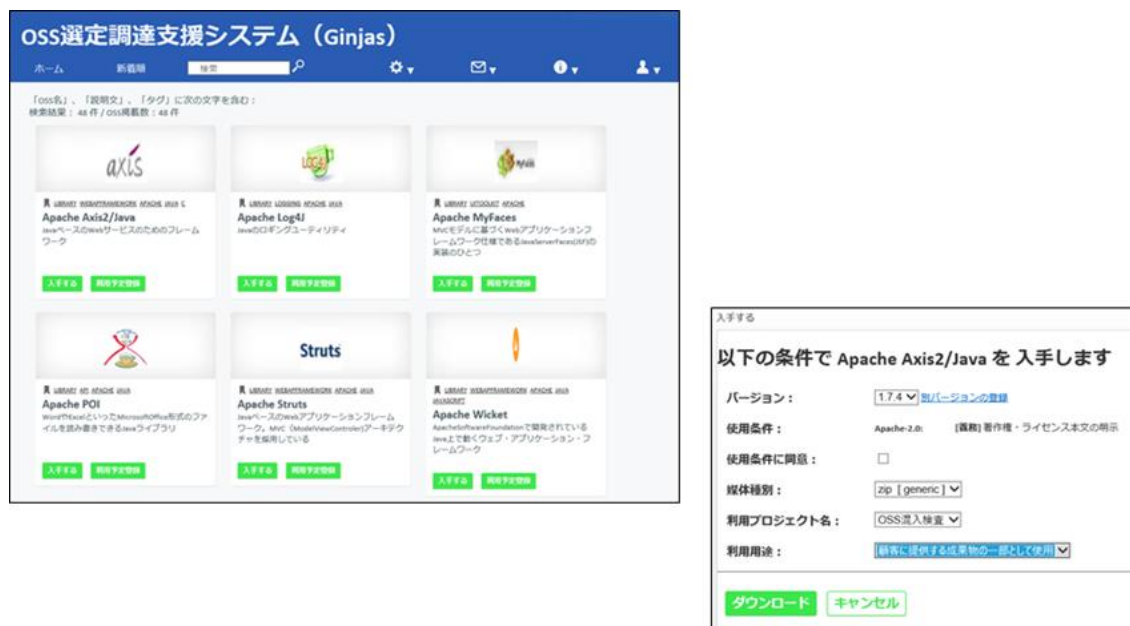


図 4.20-8 Ginjas の主な画面（登録された OSS 一覧と OSS ダウンロード）¹⁴²

Ginjas に登録している OSS の SBOM は、SCSK 独自の形式である。具体的には、OSS の種類、バージョン、ライセンス、入手先など項目を管理している（表 4.16 1 参照）。Ginjas の目的を実現するために必要な項目を管理するため SPDX Lite 等で標準化されているフォーマットは参考になっているが、SCSK 独自のフォーマットで管理している。

¹⁴² SCSK 提供資料より抜粋

表 4.20-1 Ginjas で管理している OSS の項目¹⁴³

分類	項目	内容
基本情報	名称	OSSの名称
	画像	OSSのアイコン画像
	カテゴリ	ライブラリ、フレームワーク等の分類
	ジャンル	Web開発、データアクセス等、主な用途
	ライセンス	ライセンス種別
	プロジェクトURL	プロジェクト公式HPのURL
	リポジトリ	OSSを公開しているリポジトリ
	概要	OSSの説明
	開発元	OSSを開発している企業・団体の名称
	開発元URL	開発企業・団体の公式HPのURL
	組織タイプ	企業、専用基金、個人等の種別
バージョン情報 ひとつのOSSに 複数登録	バージョン	バージョン番号、名称
	リリース日	当該バージョンのリリース日
	安定版	当該バージョンが安定版か（True/False）
	アクティブ	EOLになっておらず有効か（True/False）
	アップデート	アップデートが継続されているか（True/False）
	パッケージタイプ	Zip、tar.gz等のGinjasからのダウンロード形式
	URL	当該バージョンを公開しているURL
	プラットフォーム	Windows、Linux等の対応プラットフォーム
CPE情報	CPEベンダ	CPEベンダ名。脆弱性情報の検索用。
	CPE製品	CPE製品名。脆弱性情報の検索用。

＜【選定評価(s20)】よりよい OSS が簡単に見つけられる“物差し”があれば失敗を回避できる＞

SCSK では、情報や知識が少なくても、よりよい OSS が簡単に見つけられる“物差し”によって、無駄な失敗を回避し、OSS を効率的に活用できるツールとして OSS Radar Scope[®]を開発した。OSS の選定は、選択肢が多く、情報収集にもコストが必要になるが、OSS Radar Scope では、Web で公開されている OSS の成熟度、品質、サービス概要などの一般情報を集積し、SCSK の知見も含め、わかりやすい形式で提示することを目指した。

¹⁴³ SCSK 提供資料より抜粋

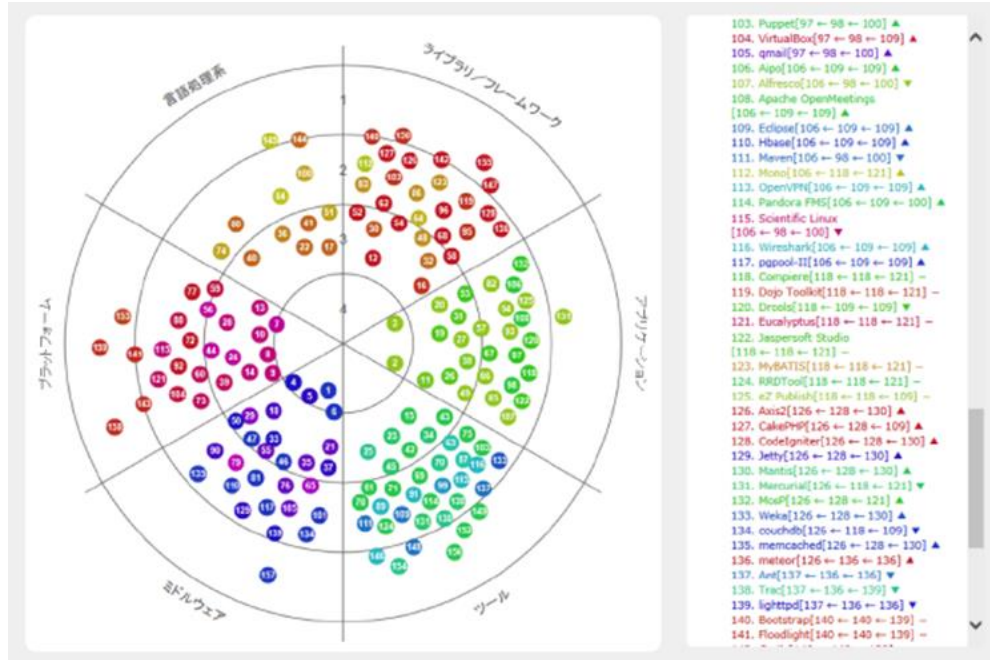


図 4.20-9 OSS Radar Scope®の構造¹⁴⁴

OSS Radar Scope®では、以下の2つの仮説に基づき、5つの視点に関する客観的なデータを収集し、月次で情報を更新し、レーティングを行っている。

仮説 1) 計画的なバージョンアップを経ている OSS は良質の OSS である可能性が高い

仮説 2) 出版された書籍の冊数が多い OSS は良質の OSS である可能性が高い

視点 1) リリース履歴

- 計画的なバージョンアップを経ている OSS は良質の OSS である可能性が高いと考えられ、パッチリリースの頻度が適切な OSS は良質の OSS である可能性が高いと考えられる。

視点 2) ドキュメント／関連出版物

- ユーザー向けドキュメントの質／量に力を入れているのは良質の OSS である可能性が高いと考えられ、また出版された書籍の冊数が多い OSS はユーザーが利用しやすく良質の OSS である可能性が高いと考えられる。

視点 3) サポート情報

- サービスを提供する「ベンダの数」×「提供されるサービスの品揃え」と OSS の質は比例すると考えられる。

¹⁴⁴ オープンソースソフトウェアへの取り組み：開発エピソードや運営の裏話

https://www.scsk.jp/product/oss/radarscope_1.html

視点 4) コミュニティの活動状況

- 開発者間でのオープンなコミュニケーションやユーザーからのフィードバックが多い OSS は良質の OSS である可能性が高いと考えられる。

視点 5) プロジェクトの継続性

- 初期リリースからの経過年数の長い OSS は成熟し、安定性と信頼性が高いと考えられる。

5. 事例（文献調査）

5.1. マイクロソフト ～OSSに係るセキュリティリスク緩和策～

■ 企業情報・本事例集における位置づけ

本社	米国
業種	情報・通信業
従業員数	163,000 人（2020 年 8 月時点）
商流	SIer 商流/サービス事業者
観点	「脆弱性対応」

■ 取組

<【脆弱性対応(v20)】OSS を利用する際のセキュリティリスク緩和策>

マイクロソフトは、OSS を利用する際、以下 4 つのセキュリティリスク緩和策を実施している¹⁴⁵。

- ① 使用している OSS を明確にする
- ② OSS の使用状況をデータベース等で集中管理する
- ③ 使用している OSS が安全か確認する
- ④ セキュリティ面の脆弱性に対応する

以下、各取組の詳細について記載する。

¹⁴⁵ Who Wants a Thousand Free Puppies?

<https://download.microsoft.com/download/C/5/E/C5E50D16-6E0B-453A-9A69-195CF3D2C823/LocoMocoSec->

2019%20-%20Who%20Wants%20a%20Thousand%20Free%20Puppies%20-%20Michael%20Scovetta%20-%20Microsoft.pdf

① 使用している OSS を明確にする

ソフトウェアを"discover"と呼ばれるツールを用いてソースコードのスキャンすることで、使用 OSS を検知し、レポート（Bill of materials（BOM））を作成する。このプロセスは、ソフトウェア開発者の手間を削減するために高度に自動化されている。

② OSS の使用状況をデータベース等で集中管理する

①において把握された OSS の使用状況は、データベース等で集中的に管理され、いつでも参照できるようにになっている。これにより、各組織の OSS の使用状況を可視化するとともに、OSS に脆弱性等が判明した際の対応におけるレスポンスタイム向上とコスト削減を図っている（図 5.1-1）。

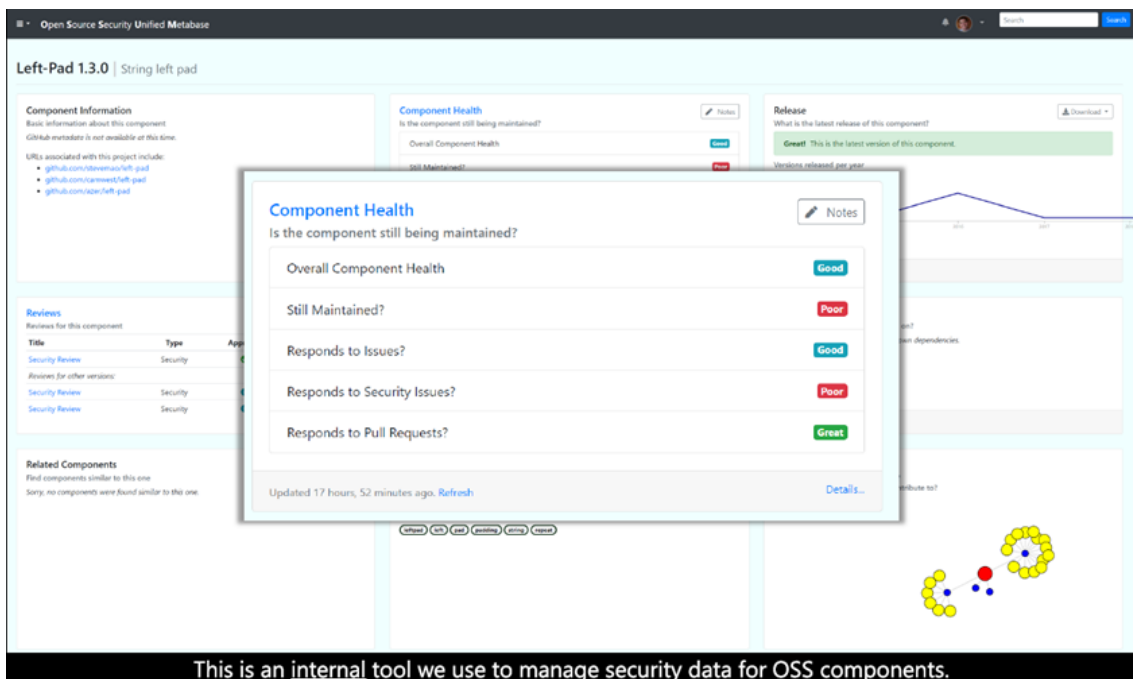


図 5.1-1 OSS コンポーネント関連データを管理する社内ツール ¹⁴⁵

（画面上に OSS コンポーネントのメンテナンス状況が表示されており、参照することで OSS に潜むリスクを把握することが可能）

③ 使用している OSS が安全が確認する

認識された OSS は、社内各組織のリスク許容度に応じて、その安全性の検証が行われる。全ての OSS コンポーネントに対して、公開情報や商用データベース情報の収集、及びツール使用を通して脆

弱性の有無が確認される。更に高いセキュリティが求められる場合は、前述のツール使用に加えてセキュリティレビュー（In-Depth Security Reviews）を実施することでその OSS コンポーネントが安全であるか確認される（図 5.1-2）。In-Depth Security Reviews は、エンジニアチームに支援されたレビューチームが携わる。コストはかかるものの、セキュリティ面における高い保証を得られる。

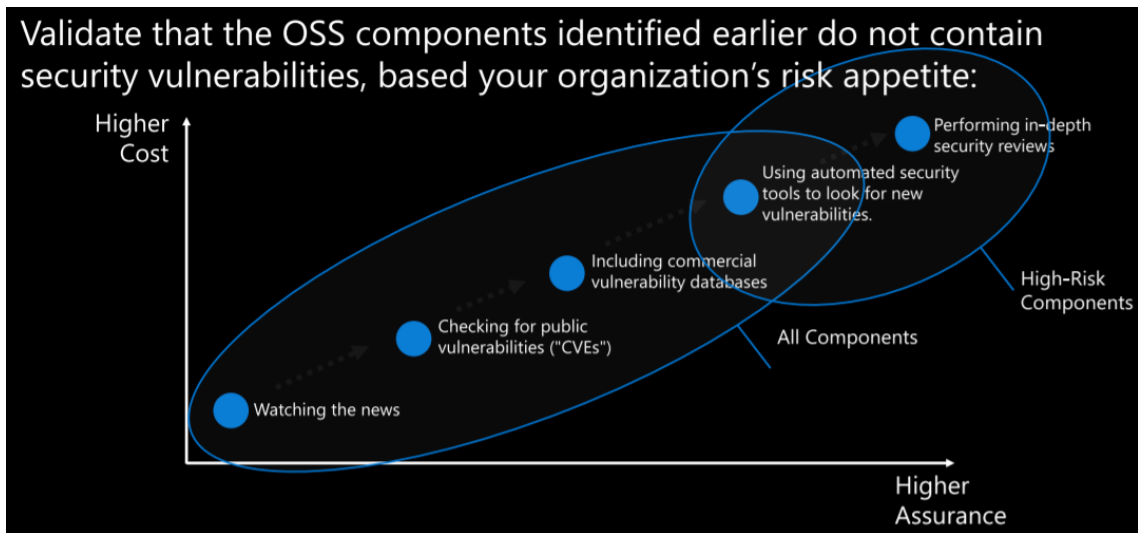


図 5.1-2 求めるセキュリティレベルとコストとのバランスに応じた OSS の安全性確認¹⁴⁵

また、その他にもセキュリティアセスメントや統計分析の活用等、様々な手段を用いて脆弱性情報の把握に努め、どの OSS が安全であるかは社内で常にアップデートされている。

④ セキュリティ面の脆弱性に対応する

マイクロソフトでは、組織的なセキュリティプログラムを運営している Microsoft Security Response Center が、OSS のセキュリティ面における脆弱性対応を統合的に実施している。Microsoft Security Response Center の特筆すべき取組として、Microsoft バグ報奨金プログラムを通じた新たな脆弱性発見のためのエコシステム構築や、判明した脆弱性やその対応策に関する情報公開等があげられる。

5.2. ザランド ～OSS プロジェクトの全社的な推進～

■ 企業情報・本事例集における位置づけ

本社	ドイツ
業種	情報・通信業（ファッション製品等の通販サイト運営）
従業員数	13,825 人（2020 年 5 月時点）
商流	SIer 商流/エンドユーザー（IT サービス）
観点	「ライセンス」

■ 取組

<【ライセンス(L21)】OSS プロジェクトの全社的な推進>

ザランドでは、OSS プロジェクトの促進、ライセンス遵守、コミュニティへの貢献を目指すため、OSS に係る以下 3 つの理念を掲げている¹⁴⁶。

- 参加（Participate）：
社員の OSS プロジェクトへの貢献を促進する。
- 共有（Share）：
技術がより多くの人の役に立つよう、コードやプロセスを共有する。
- プロモーション（Promote）：
エンジニアチームにおけるオープンソースコードの使用を奨励し、オープンソースに関するプラクティスを発信する。

また、ザランドのオープンソースチームは OSS の利活用及び開発の奨励、ベストプラクティスの共有等を目的として部門横断的な活動を展開している。具体的には、OSS に関する以下のドキュメントを作成し、社内外に展開・公表している¹⁴⁷。

¹⁴⁶ ザランドホームページ

<https://opensource.zalando.com/>

¹⁴⁷ ザランドホームページ Documentation

<https://opensource.zalando.com/docs>

- ライセンスガイド (Open Source Licensing Guide)
- オープンソースコードの採用 (Adopting open source code)
- オープンソースプロジェクトのリリース (Releasing an open source project)
- 2018 年次レポート (State of open source at Zalando 2018 Report)
- アンチハラスメントポリシー (Anti-Harassment Policy)

この内、OSS の利活用及び開発に関連するガイドラインである①「ライセンスガイド」及び②「オープンソースプロジェクトのリリース」について、以下概説する。

①ライセンスガイド¹⁴⁸

ライセンスガイドでは、使用可能なライセンス、使用を控えるべきライセンス等について説明をしている。具体的には、ザランドでは、以下 3 種類のライセンスに該当する OSS の使用を認めている。

- 許可型ライセンス (Permissive licenses) :
コピーライトとライセンス情報を付することで、使用、改変、配布が可能なライセンス。
AFL、Apache、BSD、MIT、MS-PL、ISC、PHP License 等。
- 弱コピーレフトライセンス (Weak copyleft licenses) :
コピーライト、ライセンス、変更記録、ソースコード、インストール方法の情報を付することで、使用、改変、配布が可能なライセンス。
APSL、CDDL、CPL、EPL、IPL、MPL 等。
- 強コピーレフトライセンス (Strong copyleft licenses) :
コピーライト、ライセンス、変更記録、ソースコード、インストール方法の情報、オリジナルソースを付することで、使用、改変、配布が可能なライセンス。ザランドでは社内用にのみ使用可能となっている。
BCL、GPL、LGPL、NPL、OSL、QPL 等。

一方、ザランドでは AGPL 関連ライセンス、共通条項 (Commons Clause) ライセンス及びライセンスなしのコードの使用は認めていない。

¹⁴⁸ ザランドホームページ Licensing

<https://opensource.zalando.com/docs/resources/licensing/>

②オープンソースプロジェクトのリリース¹⁴⁹

ザランドの社員は、Zalando-Incubator¹⁵⁰を通じて、新規オープンソースプロジェクトをリリースすることが可能である。プロジェクトを実施するために従うべきルールとして以下の項目を定めている。

- 既存業務の調整（Get sign off）：
プロジェクトを開始する社員は、新しいオープンソースプロジェクトに業務時間を割くことについて、所属するチームのリーダーとメンバーから承認を得る必要がある。
- コンプライアンス（Be Compliant）：
全てのオープンソースプロジェクトはザランドのルールブック（Rules Of Play）やベストプラクティスに従ってコードをリリースしなくてはならない。具体的には、ライセンス等に関連して必要となるファイルを作成することや、成果物のバージョン名を有意に設定すること等が求められる。
- レポジトリの準備（Prepare your repository）：
リリースに向けて、コードの保守性・可読性を高めたり、必要なドキュメントを準備する等、レポジトリの準備をする。また、コードが前述のルール等に準拠しているか、ザランド外の環境でも動作するか、コード内に機密情報が含まれていないか等、いくつかの観点でチェックする。
- レビューの実施（Get Reviewed）：
リリースの準備ができたコードは、オープンソースレビューグループのレビューを受けなくてはならない。レビューに合格すれば、Zalando-Incubator を通じてリリースされる。

¹⁴⁹ ザランドホームページ Releasing a new open source project

<https://opensource.zalando.com/docs/releasing/index/>

¹⁵⁰ Zalando-Incubator とは、新規オープンソースプロジェクトの立ち上げを支援するザランドの GitHub コミュニティである。ザランド社員が新規オープンソースプロジェクトをリリースする場合、Zalando-Incubator において試験的にオープンソースプロジェクトを立ち上げ、当該プロジェクトのレビューと持続可能性の評価が行われる。ザランドのオープンソースレビューグループが Zalando-Incubator のプロジェクトの評価を担い、アクティビティやコミュニティ参加状況に関して一定の水準をクリアすれば Zalando-Incubator を卒業し、ザランドのメイン GitHub コミュニティにおけるオープンソースプロジェクトとして昇格するシステムとなっている。

<https://opensource.zalando.com/docs/releasing/incubation/>

<https://opensource.zalando.com/docs/releasing/graduation/>

5.3. Linux Foundation とハーバード大学による Census II プロジェクトの予備的レポート ～アプリケーションに最も利用されている FOSS コンポーネントに関する調査～¹⁵¹

■ 本事例集における位置づけ

観点	「脆弱性対応」

■ 本事例のエッセンス

- Census II プロジェクトを通じて、現代のソフトウェアの大半を占める FOSS (Free and Open Source Software) に関し、特に、広く利用されている FOSS コンポーネントが特定された。
- FOSS の上位貢献者のうち、4 分の 3 以上が企業の従業員であることが明らかになった。
- 調査の過程で、ソフトウェアコンポーネントの標準化された命名規則の欠如、個人の開発アカウントのセキュリティの重要性の増大、OSS におけるレガシーソフトウェアの持続性という課題が明らかになった。

■ Census II の背景

現代のソフトウェアの 8～9 割は FOSS コンポーネントで構成されているとされている。しかし、FOSS の品質やメンテナンスを保証する機関が存在しないことや、自由に複製や改変が可能であることから、どのような FOSS が最も広く使われているのかは不明瞭である。

FOSS のエコシステムの将来の健全性と安全性を確保するために、民間及び公的機関でどのような FOSS が利用されているのか、それらがどの程度メンテナンスされ、サポートされているのかを理解することが非常に重要であるとの認識の下、Linux Foundation の Core Infrastructure Initiative (CII)¹⁵² とハーバード大学イノベーションサイエンス研究所により Census II プロジェクトが立ち上げられた。

¹⁵¹ Vulnerabilities in the Core Preliminary Report and Census II of Open Source Software

https://www.coreinfrastructure.org/wp-content/uploads/sites/6/2020/02/census_ii_vulnerabilities_in_the_core.pdf

¹⁵² Heartbleed 問題 (OpenSSL 脆弱性) の問題を受け、Linux Foundation により 2014 年に設立された機関

■ レポートの概要（【脆弱性対応(v22)】）

Census II プロジェクトの予備的なレポートとして、2020 年 2 月に「Vulnerabilities in the Core Preliminary Report and Census II of Open Source Software」が公表された。レポートの概要を以下に記載する。

<調査の目的>

- ① 製品アプリケーションにおいて最も一般的に使われている FOSS コンポーネントを特定すること
- ② それらの FOSS コンポーネントについて、以下に起因する潜在的な脆弱性を調査すること
 - 古いバージョンの広範な利用
 - プロジェクトの人員不足
 - セキュリティ上の脆弱性が知られていること
- ③ これらにより得られた情報を、FOSS の健全性と安全性をサポートするための投資における優先順位付けに活用すること。

<調査結果>

パートナーであるソフトウェア構成分析をする企業やアプリケーションのセキュリティ企業から提供を受けたデータセットについて依存関係の分析を実施し、その中で最も利用されている FOSS コンポーネントを明らかにした。レポートでは、付録として最も使用頻度の高い 10 のパッケージがアルファベット順に列挙されている（表 6）。なお、JavaScript パッケージが圧倒的に多かったため、JavaScript パッケージを除外した場合の上位パッケージを特定するために、JavaScript と JavaScript 以外のそれぞれについて抽出されている。ただし、非推奨であったり、数年間更新されていないいわゆる「レガシーソフトウェア」が利用されている場合に、それが現在も積極的に使われているのかについては更なる深い調査が必要であることや、調査に用いられたデータセットは量的にも質的にも限定的なものであったことから、予備的レポートの調査結果は、どの FOSS パッケージが最も重要であるかを主張するものではない旨、補足されている。

表 5.3-1 最も利用されている FOSS パッケージ

JavaScript パッケージ	JavaScript 以外のパッケージ
async	com.fasterxml.jackson.core:jackson-core
inherits	com.fasterxml.jackson.core:jackson-databind
isarray	com.google.guava:guava

JavaScript パッケージ	JavaScript 以外のパッケージ
kind-of	commons-codec
lodash	commons-io
minimist	httpcomponents-client
natives	httpcomponents-core
qs	logback-core
readable-stream	org.apache.commons:commons-lang3
string_decoder	slf4j:slf4j

また、FOSS の開発者について、上位貢献者のうち、企業に雇用されている者として特定された者は 75%、個人事業主として特定された者は 15%（残り 10%は不明）であり、企業の従業員の割合が高いことが明らかとなった。2017 年の GitHub のデータを分析したところ、最もアクティブな FOSS 開発者の中には、マイクロソフト、グーグル、IBM、インテルの従業員用のメールアドレスでプロジェクトに貢献している者がいることが判明した。付録に記載されているプロジェクトへの貢献者が、これらのパッケージ開発のために雇用元の企業から直接報酬を受け取っていないとしても、FOSS コミュニティのメンバーとしての彼らの地位が、雇用のための資格を裏付けるものであった可能性があることを指摘している。ただし、貢献者がどのような状況で活動しているのかをより明確にし、それらの仮説を裏付ける直接的なデータがなければ結論を出すことはできない旨、補足されている。

<調査によって得られた課題>

Census II プロジェクトの初期段階に、当プロジェクトの取組とは別に、以下の課題が明らかになった。レポートでは、これらの課題は、調査の結果に影響を与えるものではないとしつつ、より広範な議論がされるべき重要なものであり、調査に値するものであるとしている。

① ソフトウェアコンポーネントの標準化された命名規則の欠如

ソフトウェアコンポーネントの命名規則が標準化されていないため、データセットを分析するためのデータセットの統合に際して多大な労力が必要とされた。NIST¹⁵³や NTIA のプロジェクトでも明らかとなったように、ソフトウェアサプライチェーンの透明性とセキュリティへの取組が発展し、より複雑になるにつれて、命名規則の欠如は、産業界や政府によるソフトウェアベースのインシデントから自身を守るための取組

¹⁵³ National Institute of Standards and Technology : アメリカ国立標準技術研究所

を阻害する脅威となっていることを指摘している。

② 個人の開発アカウントのセキュリティの重要性の増大

最も利用されているソフトウェアパッケージの上位 10 件のうち、7 つのパッケージが個人の開発者アカウントにより開発されているなど、多くのプログラムが個人の開発アカウントに存在することが明らかになった。個人の開発アカウントは、企業のアカウントと比べて脆弱である可能性があるため、個人の開発アカウントの管理下にあるコードの変更はかなり容易に、しかも検出されずに行うことができる旨指摘されている。これはリスクに留まるものでなく、実際に発生しており、悪意ある者が正当な管理権限を委譲されてバックドアを仕掛けた Copay¹⁵⁴の事例は、厳密にいうと乗っ取りではないが、個人の開発アカウントは侵入や乗っ取りの危険性もあることが指摘されている。また、Left-pad¹⁵⁵の事例では、パッケージの命名争いを発端として開発者がコードをレポジトリから削除したことにより、たとえそれが一見些細なコードであるように見えても、当該コードに依存していた多くのパッケージが機能しなくなったなど、開発者本人の行動が大きな影響を及ぼした例も挙げられている。

③ OSS におけるレガシーソフトウェアの持続性

基本的に同じ機能を有する新しいパッケージが存在するにもかかわらず、古いパッケージの利用率の方が高いケースがある。互換性のバグへの懸念や、改修にかかる時間やコストの制約から、新しいソフトウェアへの切り替えが進みにくいことが原因と想定されている。古いパッケージの開発者は時間とともに減少するため、FOSS のレガシー問題についても認識する必要がある旨指摘されている。

¹⁵⁴ bitpay 社が手掛けるモバイルウォレットアプリ

¹⁵⁵ 文字列の調整を行う機能を持つ JavaScript で書かれたライブラリ

6. まとめ

本事例集では、ヒアリングにより収集した国内企業の事例（20 事例）と、公開文献の調査により収集した海外事例（3 事例）を紹介した。本章においては、3.2.2 章において取りまとめた観点ごとに各事例における取組のポイント等を取りまとめる。

■ 選定評価

OSS の選定評価においては、選定基準の明確化が重要視されていた。例えば、使用可能もしくは使用不可な OSS をリスト化し、社内もしくはグループ内の OSS 選定においてそのリストへ準拠することをルール化する手法を採る企業や選定調達を支援するシステムを構築している企業が見られた。その他、使用 OSS に関するポリシーを定め、そのポリシーに則り OSS を選定することをルール化する方法も採られていた。さらに、よりよい OSS が簡単に見つけられる物差しを独自に定め、物差しに基づき情報収集し、一般公開する取組もあった。

これらのリストやポリシーが常に最適な OSS の使用基準となるよう、専門的な知識を持ったメンバーによる定期的なレビューやメンテナンス等の工夫が見られた。また、各 OSS の使用可否の判断やポリシーの設定においては、後述の「ライセンス」「保守、品質管理」等の観点や、当該 OSS の社内もしくはグループ内の使用実績、コミュニティの活動状況等が考慮されていた。

■ ライセンス

ライセンス対応は、特に製品商流において重点的に取り組まれており、OSS 管理に関する取組を開始するきっかけとしてライセンスコンプライアンスをあげる企業も多かった。その対応においては、OSS を使用する際に当該 OSS のライセンスの内容を組織として把握し、対応要件を遵守可能か等の観点から審査する手法が採られていた。あわせて、一連のワークフローのシステム化、ツール等を用いた未認識の OSS の有無の確認、特定の開発段階での承認のルール化等、確実なライセンス対応を担保するための工夫も見られた。ワークフローシステムに使用 OSS の情報を蓄積し、後述の「脆弱性対応」において活用している事例もあった。

また、ライセンス対応においては法務の知見が必須となる。ワークフロー上に法務部門がレビューする機会を設けたり、必要に応じて法務部門が事業部にアドバイスをを行う等の対応が見られた。

さらに、ライセンス管理や構成管理として、ビルドシステム等の OSS に備わっている機能を活用しながら、ソフトウェアの依存関係を管理している取組や検証ツールの自動化による早期検証や省力化が考慮されていた。

■ 脆弱性対応

脆弱性対応においては、SBOMを活用する企業が多く見られた。SBOMの作成は、前述のライセンス対応で得られた使用 OSS 情報の活用や、ツールの利用等により行われていた。収集した OSS の脆弱性情報を SBOM と突合して影響を把握し、必要に応じて当該 OSS を使用している製品やサービスの担当部署によって適切な対応がなされていた。OSS の脆弱性情報の収集は、JVN 等からの情報に加え、一般的な情報ソース（ニュースサイトやブログ等）からも収集することで、より迅速な対応を可能にするといった工夫も見られた。複数の企業で、脆弱性のハンドリング対応において中心的な役割を果たす PSIRT 等の組織を設け、各部署と円滑に連携して対応する体制が採られていた。

他方、SBOM で管理している情報と収集した脆弱性情報における、OSS 名称や情報粒度に差異があるため、適切に突合ができないという課題も認識されていた。一部企業では、AI 等の先進技術を用いて解決を模索していた。

■ 保守、品質保証

保守、品質保証においては、SBOM 等を含む OSS 管理の仕組みの中で、EOL の時期も管理することで、適切な対応が実施されるよう工夫する企業が見られた。また、保守可能なディストリビュータを介した OSS の調達や、社内人材でワークアラウンドできる OSS の使用をルール化するという手法も採られていた。

社会インフラ系のシステム等に使用される OSS においては、その分野的特性から長期保守を実現させるための議論も実施されている。また、通信事業では、長期的な OSS の活用におけるリスクとして、特に脆弱性管理やアップデート対応に関するコスト等について、予め顧客と SIer が事前に合意する取組があった。

■ サプライチェーン管理

サプライチェーン管理においては、サプライチェーンの各社で使用されている OSS を把握するために、サプライヤ企業や委託先との取決めや契約の中で、使用 OSS の申告を義務付けている企業が見られた。そうした取決め・契約の締結の際には、サプライヤ企業や委託先からの理解を得るため、サプライチェーンにおける OSS 使用状況把握の必要性を丁寧に説明する等、地道な努力が行われていた。

また、特筆すべき活動として、OpenChain の自己認証取得や OpenChain を通じた情報共有・発信が挙げられ、ライセンスコンプライアンスの重要性に関する共通認識の醸成に取り組む企業が多く見られた。このような企業の壁を越えた取組が、サプライチェーン全体の OSS に対するリテラシー向上に寄与している。

■ 個の能力、教育

社員の OSS に対するリテラシー向上のために教育メニューを整備している企業が見られた。具体的には、OSS の利活用啓発、OSS を利用した開発の進め方、ライセンスコンプライアンス対応等について多彩な教育メニューが用意されていた。社員に高いリテラシーを身に付けさせるため、一定の得点を目標とした試験を実施する工夫も見られた。また、海外拠点においても国内の社員と同等レベルのリテラシーを確保するために、英語教材を整備している企業もあった。

さらに、経営層のキーマンが理解を示したことで、OSS の取組が加速された企業が複数あった。このことから、OSS 利活用を進めるためには、現場レベルのリテラシー向上だけでなく、経営層に対する啓発や情報共有によって理解を得ることが重要であると推察される。

■ 組織体制

OSS 対応に係る組織体制に関しては、OSS 対応を主なミッションとする専門組織を持つ企業がある一方、各部門から代表者が集まり部門横断的でコミュニティのような対応組織を持つ企業がある等、様々な形が見られた¹⁵⁶。さらに、OSS の利用可否の意思決定を効率的に行うため、開発部門と OSS 管理部門のリーダー同士がリスクの確認や懸念点を議論して決定する取組もあった。また、セキュリティ対応を担う部署の社員が OSS 対応を兼任する体制も、OSS に対する取組が発展途上である場合に多く見られた。

OSS 対応には、OSS 自体や OSS を利活用している自社の製品・サービスの十分な理解に加え、セキュリティや法務、調達、品質管理といった複数部門にまたがる広い知見が求められる。また、製品・サービスの事業環境によっても適切な体制は異なるため、各企業は、自社の各製品・サービスの事業環境も考慮したうえで、企業内、グループ内の各部署と適切に連携する体制の構築に努めていた。

■ コミュニティ活動

OSS の利活用においては、コミュニティ活動への積極的な参加が重要な意義を持つと認識している企業が見られた。そのような企業では、社員のコミュニティ活動を業務の範囲として扱ったり、コミュニティ参加に関するガイドラインを整備する等、社員のコミュニティ活動への参加を後押しし、所属企業の利活用に問題がない場合は、社員が個人としてコミュニティに還元することも認める取組もあった。その理由は、OSS の場合、コミュニティ活動が活発化することでその性能や品質が向上するため、社員のコミュニティ活動への参加・貢献が、その OSS を利用する企業側のメリットにつながるためである。さらに、社員がコミュニティ活動において

¹⁵⁶ 経済産業省で取りまとめた「サイバーセキュリティ体制構築・人材確保の手引き」に整理されているセキュリティ統括機能の形態である「専門組織型」「委員会型」が存在していると捉えられる。

<https://www.meti.go.jp/press/2020/09/20200930004/20200930004-1.pdf>

最大限の成果をあげられるよう、コミュニティ型開発の進め方に関する研修メニューを用意している企業や、自社開発したソフトウェアを OSS 化し、コミュニティ型開発による性能向上を図る企業等、OSS のエコシステムを戦略的に活用する工夫も見られた。

二次利用未承諾リスト

令和3年度 サイバー・フィジカル・セキュリティ対策促進事業
SBOMを導入・活用するサプライチェーンモデルの構築に向けた調査・実証事業 報告書

令和3年度サイバー・フィジカル・セキュリティ対策促進事業（SBOMを導入・活用するサプライチェーンモデルの構築に向けた調査・実証事業）

株式会社三菱総合研究所

頁	図表番号	タイトル
49	図2-8	sigstoreのアーキテクチャ
54	図2-10	フランスの公的機関によるOSSレポジトリを集めたWebサイトの概要
81	図4-2	NIST公式サイト上のOpenSCAP実装の認証状況
90	図4-4	Wordpressに関連するCVE発行数の統計
92	図4-5	JPCERT/CCからみた脆弱性ハンドリング
95	図4-6	CNAの構造
100	図4-7	OSVのワークフロー
103	図4-8	米国政府脆弱性情報管理の概略（案）