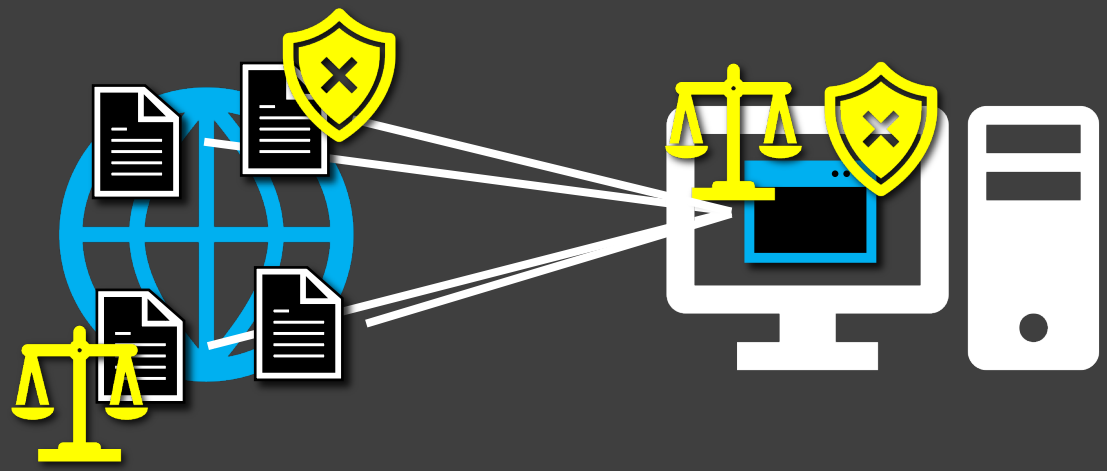
SBOM 利活用に関する課題抽出及び C/C++ のソフトウェアに対する SBOM 生成手法の開発

肥後研究室

M2 音田渉

背景： サプライチェーンリスク

- 現在のソフトウェア開発では、多くの外部ライブラリを活用
  - 開発費用の削減、開発期間の短縮、堅牢化・高機能化
  - 96% のコードベースがオープンソースのコードを含む [1]
- しかし、リスクを伴う
  - **サイバーセキュリティ（サプライチェーン攻撃）**
    - XZ Utils を経由した SSH バックドアの埋め込み（2024） [2]
  - **著作権法（ライセンス違反）**
- **依存関係の適切な管理が重要**



[1] Black Duck Software, ”2024 Open Source Security and Risk Analysis Report,” 2024
[2] L. Collin, “XZ Utils backdoor,” 2024

Software Bill of Materials (SBOM)

- ソフトウェア部品の機械可読な表
  - 部品： ライブラリなど
- **サプライチェーンリスクへの迅速な対応に貢献**
  - Executive Order on Improving the Nation’s Cybersecurity
    - US 大統領令, 2021
  - Cyber Resilience Act
    - EU 法, 2024（2027 年から適用）
  - ソフトウェア管理に向けた SBOM の導入に関する手引 ver. 2.0
    - 日本 経済産業省, 2024

SPDX®

CycloneDX

課題： SBOM の普及は不十分

SBOM の普及は未だ不十分

- 組織への準備度調査 (Linux Foundation) [3]
- 組織へのインタビュー調査、アンケート調査 [4, 5, 6, 7]
- GitHub 調査
  - OSS への普及状況の定量調査 [8]
  - SBOM 関連プロジェクト上の議論の調査 [9]

一般の開発者が実際に直面した課題の調査は未実施

[3] S. Hendrick and J. Zemlin, “The State of Software Bill of Materials (SBOM) and Cybersecurity Readiness,” 2022
[4] B. Xia ら, “An Empirical Study on Software Bill of Materials: Where We Stand and the Road Ahead,” ICSE '23
[5] T. Stalnaker ら, “BOMs Away! Inside the Minds of Stakeholders: A Comprehensive Study of Bills of Materials for Software Systems,” ICSE '24
[6] S. Nocera ら, “If it’s not SBOM, then what? How Italian Practitioners Manage the Software Supply Chain,” ICSME 2024
[7] B. Kloeg ら, “Charting the path to SBOM adoption: A business stakeholder-centric approach,” ASIA CCS '24
[8] S. Nocera ら, “Software Bill of Materials Adoption: A Mining Study from GitHub,” ICSME 2023
[9] T. Bi ら, “On the Way to SBOMs: Investigating Design Issues and Solutions in Practice,” ACM TOSEM Vol 33, No 6 (2024)

Stack Overflow 上の質問内容の調査

質問とその回答を投稿・検索できる開発者向け Q&A サイト

調査 1： SBOM 利活用に関する質問の回答・解決状況

質問の解決状況は極めて低い水準であり、SBOM 利用者が課題に直面した際に Stack Overflow で質問することでは解決が難しい状況

調査 2： SBOM 利活用に関する質問数の推移

SBOM 利用を求める大統領令の発令や SPDX の ISO/IEC 標準化などがなされた 2021 年以降、新規質問数の増加が加速

調査 3： SBOM 利用者が抱いている課題

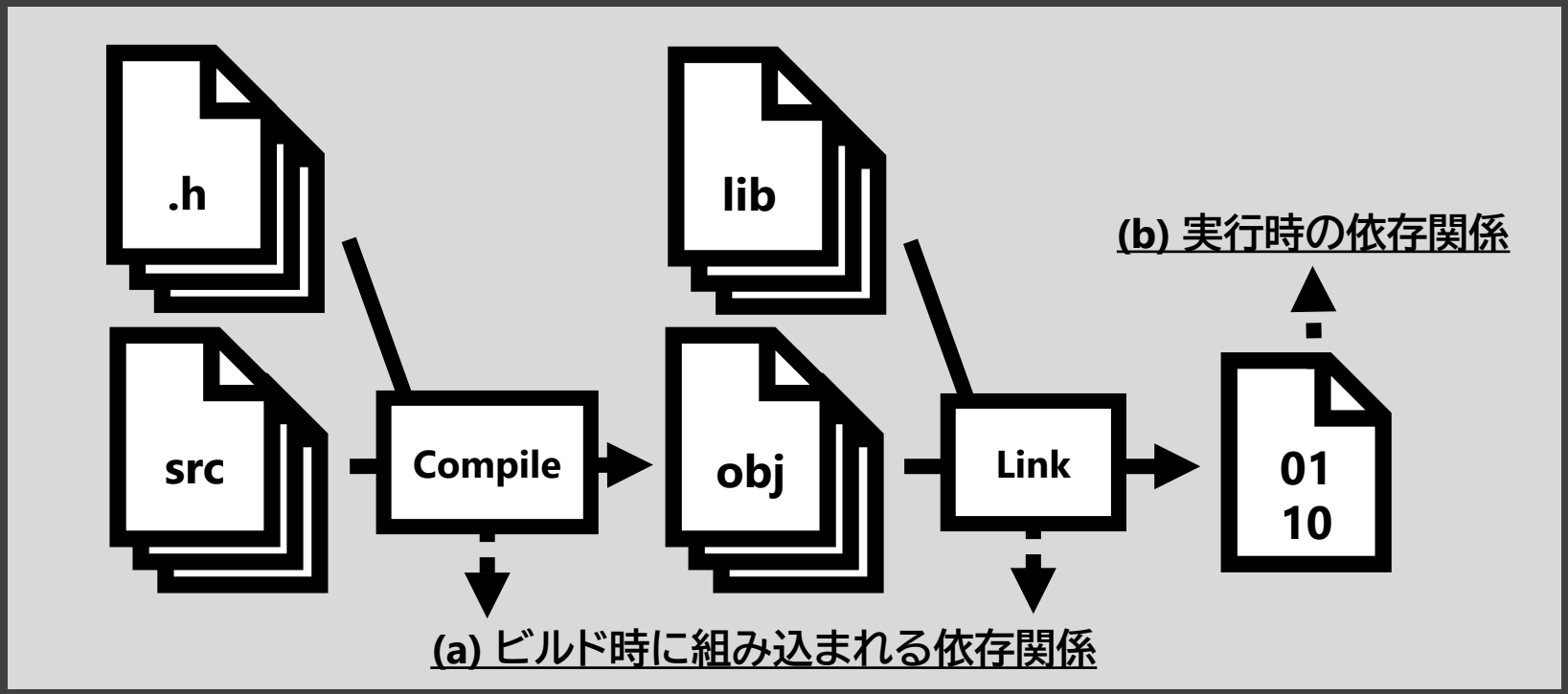
- **SBOM ツールのユースケースの網羅性不足**
- **各種要件を満たす SBOM を生成できない**
- SBOM ツールに不具合がある、または利用方法が不明瞭

C/C++ のソフトウェアに対する SBOM 生成

- パッケージマネージャの情報を用いる SBOM ツールは存在
  - 依存関係とメタデータが集権管理されている
  - npm, pip などに対応
- **C/C++ を対象に SBOM を生成する手法は存在しない**
  - パッケージマネージャが使われない傾向にあるため
  - 既存システムへのパッケージマネージャ導入は高コスト
  - 千差万別なビルドファイルの安定的な解析も困難
- GitHub 上で共に Top 10 に入る言語 [11]
- 他言語でも C/C++ 部品に依存するものが多くある
  - Python の数値計算ライブラリ NumPy など

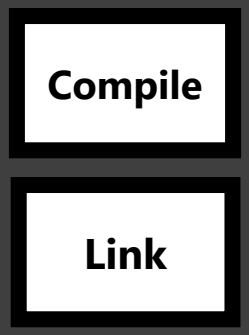
[11] GitHub Staff, “Octoverse: AI leads Python to top language as the number of global developers surges,” 2024

C/C++ のビルド処理



- コンパイル・リンク処理に着目
  - ヘッダファイルを取り込む
  - バイナリライブラリ と プログラムが使うシンボル を照合する
  - 動的(共有)ライブラリについては依存情報をバイナリに埋め込む

依存関係の抽出

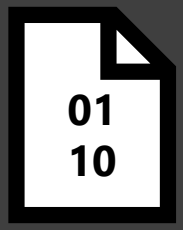


ビルド処理

(a) ビルド時に組み込まれる依存関係の抽出

コンパイル・リンク処理の内部情報を解析
静的リンクと動的リンクを取得可能
（動的リンクは必ずしも実行環境のバージョンと一致しない）

```
. /usr/include/openssl/ssl.h
/usr/lib/gcc/x86_64-linux-gnu/13/../../../x86_64-linux-gnu/libcrypto.so
```

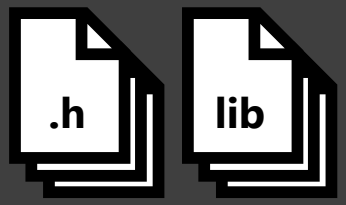


実行ファイル

(b) 実行時の依存関係の抽出

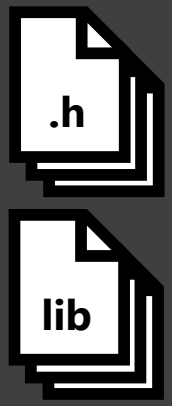
実行ファイルのヘッダ領域を解析
動的リンクのみ抽出可能
（該当環境で実際に使われるバージョンを得られる）

```
libcrypto.so.3 => /lib/x86_64-linux-gnu/libcrypto.so.3
(0x00007f411a280000)
```



依存ファイル群

依存パッケージ群の抽出（Debian 系）



依存ファイル群

ファイルパスの正規化

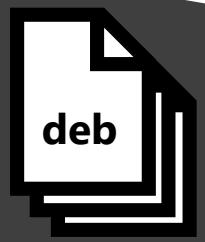
実在するファイルに書き換え
パス中の `.` `..` `//` やシンボリックリンクの解決
重複を除外

```
/usr/include/openssl/ssl.h
/usr/lib/x86_64-linux-gnu/libcrypto.so.3
```

パッケージへの対応付け・バージョン特定

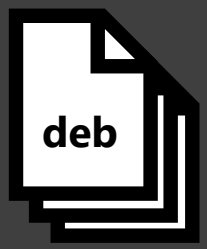
`dpkg-query` に問い合わせ
重複を除外

```
libssl-dev:amd64=3.0.13-0ubuntu3.4
libssl3t64:amd64=3.0.13-0ubuntu3.4
```



依存パッケージ群

メタデータの抽出（Debian 系）



依存パッケージ群

OS バージョンの抽出

`/etc/os-release` ファイルを解析
PURL(識別子)の生成に必要

パッケージメタデータの抽出

`apt-cache` と `apt-get` で問い合わせ
著作権情報以外のメタデータを抽出できる

著作権情報の抽出

`/usr/share/doc/パッケージ名/copyright` ファイルを解析
ライセンス式の記法が SPDX 規格と異なるため、
構文解析して対応表とヒューリスティックで変換

SPDX Document (SBOM)

評価 – 前提

- SPDX はフォーマットを規定するのみで、含める内容は自由
  - しかし、実際に運用するためには一定の基準が必要
- **NTIA Minimum Elements** [12]
  - 米国の政府機関 NTIA が大統領令に基づき発行した文書
  - SPDX 公式が準拠度確認ツールを公開している

項目	概要
Supplier Name	部品を作成・定義・特定した人
Component Name	供給者によって部品に付けられた名前
Version of the Component	旧版との区別のために供給者が設定した識別子
Other Unique Identifiers	部品の特定や DB 検索キーに使われる識別子 例： CPE, SWID tag, **PURL**
Dependency Relationship	上流部品 X がソフトウェア Y に含まれるなどの関係
Author of SBOM Data	SBOM データを作成した人
Timestamp	SBOM データを作成した日時

[12] The United States Department of Commerce, “The Minimum Elements For a Software Bill of Materials (SBOM)”, 2021

評価 – 内容

対象

- 依存関係とメタデータの抽出能力
- 生成される SBOM の SPDX 規格や NTIA 要件への準拠度

内容

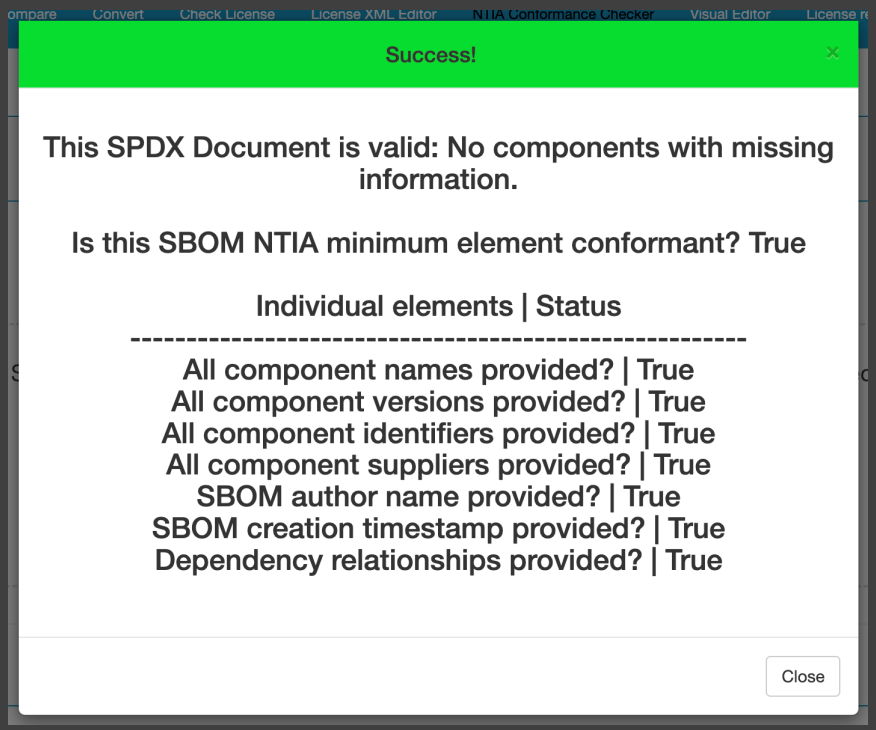
- 実際の C/C++ のソフトウェアに対する SBOM 生成
- システム上の全パッケージに対する SBOM 生成
  - メタデータ抽出処理の汎用性を詳しく検証
- SPDX Validator[1] と NTIA Conformance Checker[2] の適用
  - SPDX 公式が提供するツール

[1] https://tools.spdx.org/app/validate/

[2] https://tools.spdx.org/app/ntia_checker/

評価 – 結果

- 抽出した全ての依存関係について出自パッケージを特定
  - 依存ライブラリが正しく出力に含まれることを確認
- 生成した全ての SBOM が SPDX 規格と NTIA 要件に準拠
- NTIA 要件外の項目も高い水準で抽出成功
- 開発者の作業を妨げない実用的な実行時間



評価 – curl に対する SBOM 生成

ビルド時 – 27.9 秒

- 依存ファイル： 316 件
- 依存パッケージ： 9 件
  - `libssl-dev` と `libssl3t64` を含む

実行ファイル – 14.0 秒

- 依存ファイル： 31 件
- 依存パッケージ： 27 件
  - `libssl3t64` を含む

項目	ビルド時	実行時
name *	**100% (9)**	**100% (27)**
versionInfo *	**100% (9)**	**100% (27)**
supplier *	**100% (9)**	**100% (27)**
externalRefs (PURL) *	**100% (9)**	**100% (27)**
packageFileName	100% (9)	100% (27)
downloadLocation	100% (9)	100% (27)
homepage	88.9% (8)	100% (27)
summary/description	100% (9)	100% (27)
checksums	100% (9)	100% (27)
copyrightText	33.3% (3)	70.4% (19)
licenseConcluded/Declared	88.9% (8)	100% (27)

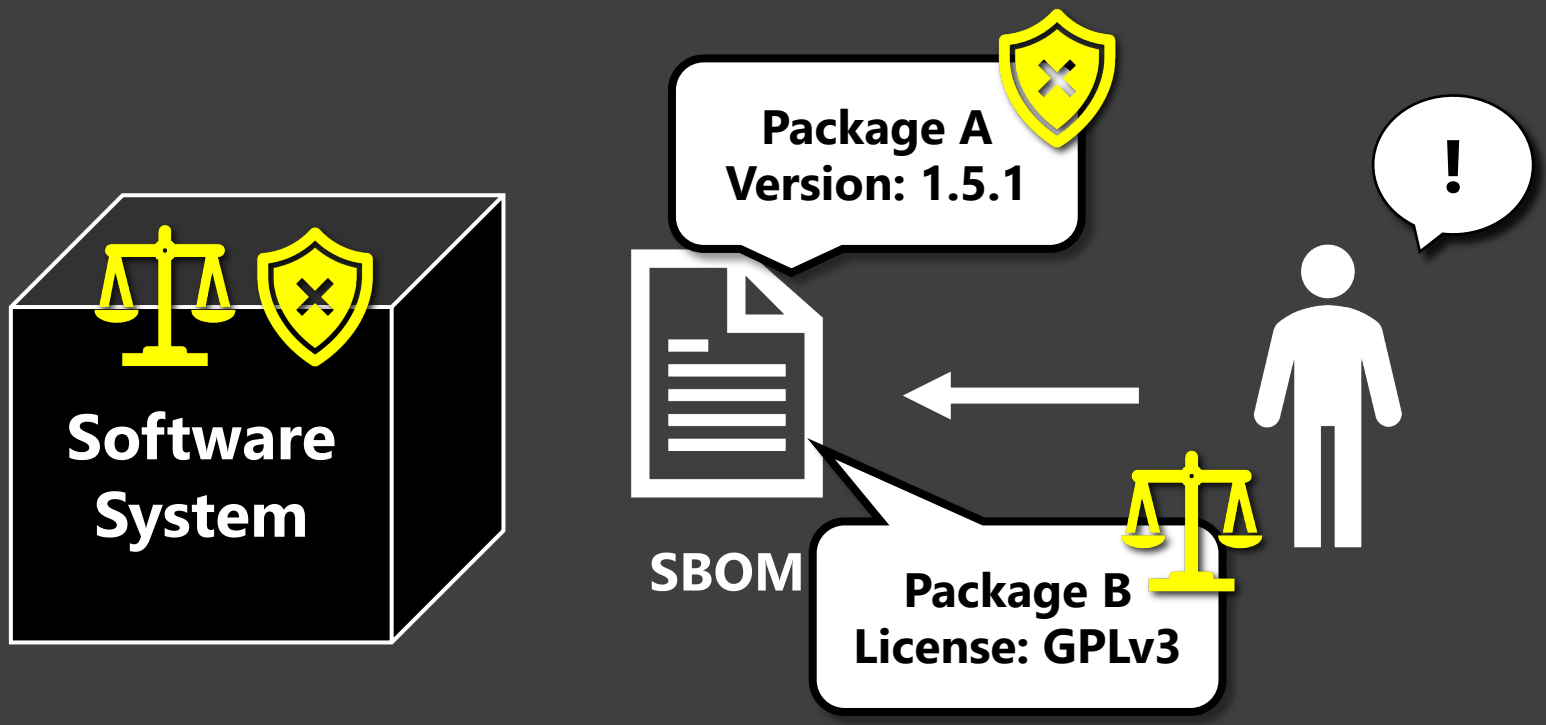
まとめ

- サプライチェーンリスク対策のため SBOM の利用が奨励
  - しかし普及にはまだ課題が多い
- **Stack Overflow 質問分析**から開発者が抱く課題を抽出
  - SBOM ツールのユースケースの網羅性不足
  - 各種要件を満たす SBOM を生成できない
  - SBOM ツールに不具合がある、または利用方法が不明瞭
- **C/C++ のソフトウェアに対する SBOM 生成手法**を開発
  - ビルド時に組み込まれる依存関係
  - 実行時の依存関係
- **NTIA 要件を達成する SBOM の自動生成に成功**
  - 実際の C/C++ のソフトウェアに対する SBOM 生成
  - システム上の全パッケージに対する SBOM 生成
  - SPDX Validator と NTIA Conformance Checker を通過

付録： SBOM の利点

ソフトウェアに含まれる部品情報を容易に得られる

→ **脆弱性・ライセンスなどの問題を迅速に把握・対応できる**



付録： 既存調査

SBOM の普及は未だ不十分

準備度調査 (Linux Foundation, 2022) [2]

多種多様な業種・規模の 412 組織を対象に、
各側面から SBOM 利活用への準備度を調査

アンケート調査 (Xia ら, 2023) [3]

オンラインで募った実践者にアンケートを行い、
SBOM への見方や普及への目標を提示

インタビュー調査 (Stalnaker ら, 2024) [4]

ステークホルダーが SBOM の作成時や使用時に
直面した課題と、課題への対処法を調査

利用状況調査 (Nocera ら, 2023) [7]

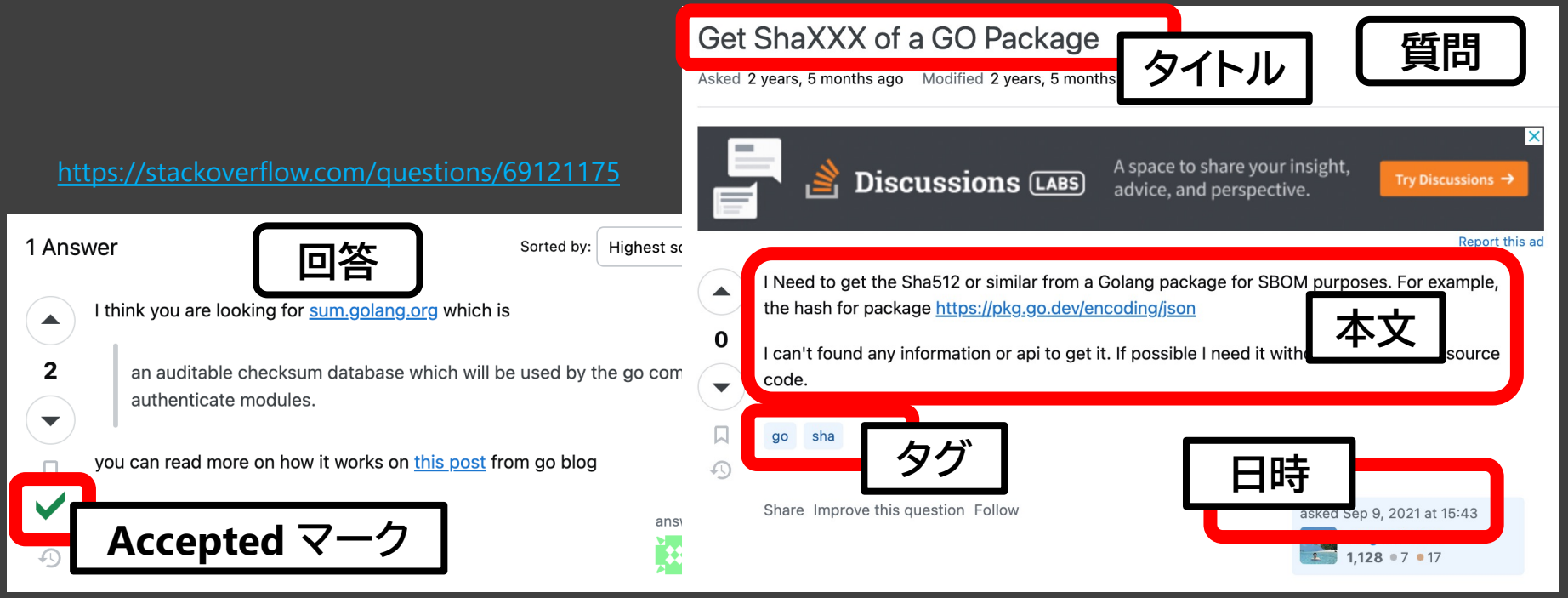
GitHub 上の OSS に対する定量調査

一般の開発者が実際に直面した課題の調査は未実施

付録： Stack Overflow [10]

- 質問とその回答を投稿・検索できる開発者向け Q&A サイト
  - **一般の開発者が、直面した課題について質問**
- ユーザ数： 約 2,700 万人
- 投稿数： 約 6,000 万件
- 新規質問数： 1 日あたり約 1,500 件

https://stackoverflow.com/questions/69121175



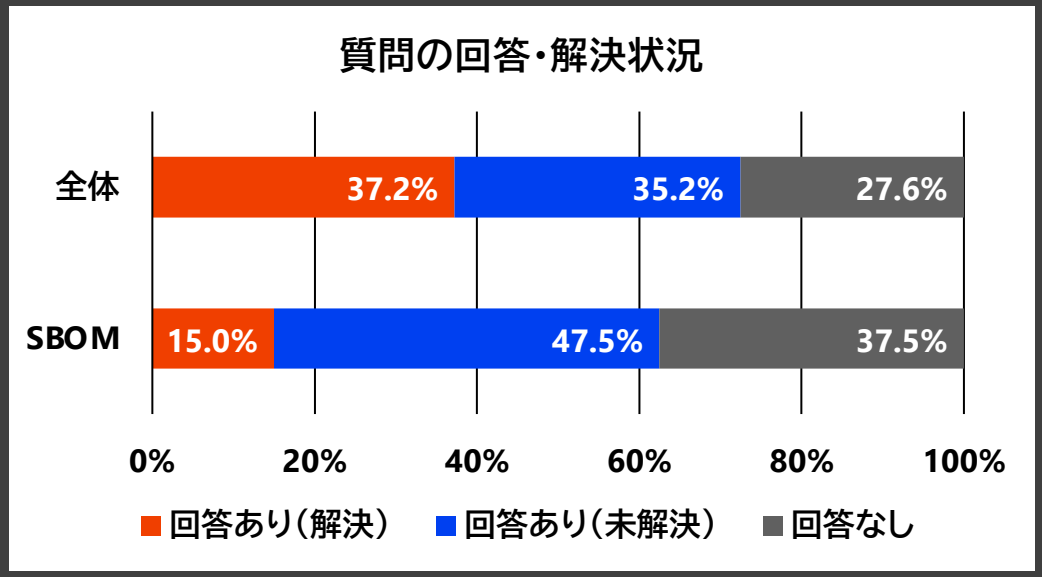
[10] Stack Exchange, “All Sites”, 2024

付録： 質問調査 1 – 回答・解決状況

- SBOM 利活用に関する質問のうち、
  - 回答あり（解決）： 6 件
  - 回答あり（未解決）： 19 件
  - 回答なし： 15 件
  - （2021 年以降）

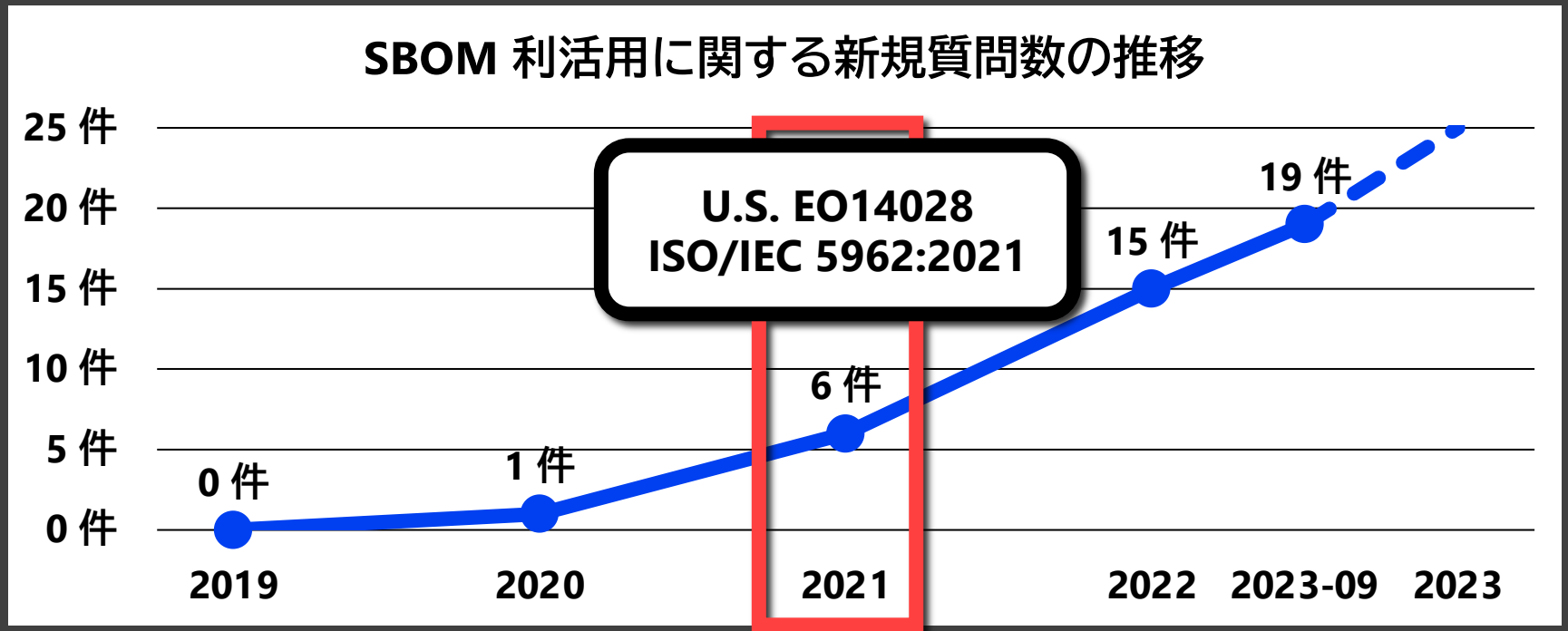
回答状況： 平均的

解決状況： 低い



- **SBOM の課題を Stack Overflow で解決することは困難**
  - 知見が不十分で、質問に回答できるソフトウェア開発者が不足
  - 技術が未熟で、質問時点では技術的に解決不可能な事項が多い

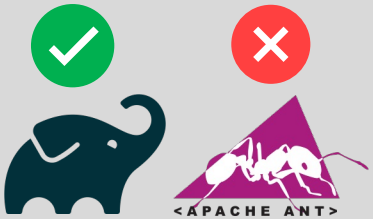
付録： 質問調査 2 – 質問数推移



- 4 年間を通して少数
  - これ以前は 2012 年の 1 件のみ
- 2021 年頃から増加
  - 米国大統領令や SPDX の ISO/IEC 標準化などの影響？

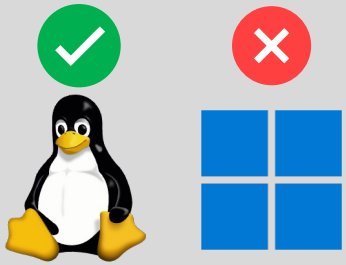
付録： 質問調査 3 – 課題 1

SBOM ツールのユースケースの網羅性不足



旧式のプロジェクト管理システムへの対応が不十分

“How do I generate a Cyclonedx bom for a Java project built with Ant?” [1]



Windows への対応が不十分

“Is there any tool through which we can generate SBOM report (SPDX / CycloneDX) for Windows programs?” [2]

[1] https://stackoverflow.com/questions/71605182 (March 2022)
[2] https://stackoverflow.com/questions/73648096 (September 2022)

付録： 質問調査 3 – 課題 2

各種要件を満たす SBOM を生成できない

ツールが NTIA のガイドラインに追従しきれていないか、ツールの実装において特定項目の入手が困難

“NTIA minimum SBOM requirement tool” [1]

（2021 年 7 月に、NTIA が米国大統領令に伴い発行）

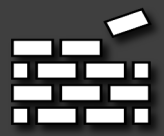
GitHub の SBOM 生成機能は、ソフトウェアライセンスコンプライアンス管理目的では不足

“How to include Open Source license in GitHub SBOM export?” [2]

（リポジトリから SPDX 形式の SBOM を自動生成する機能）

[1] https://stackoverflow.com/questions/76103711 (April 2022)
[2] https://stackoverflow.com/questions/76962834 (August 2023)

付録： 質問調査 3 – 課題 3

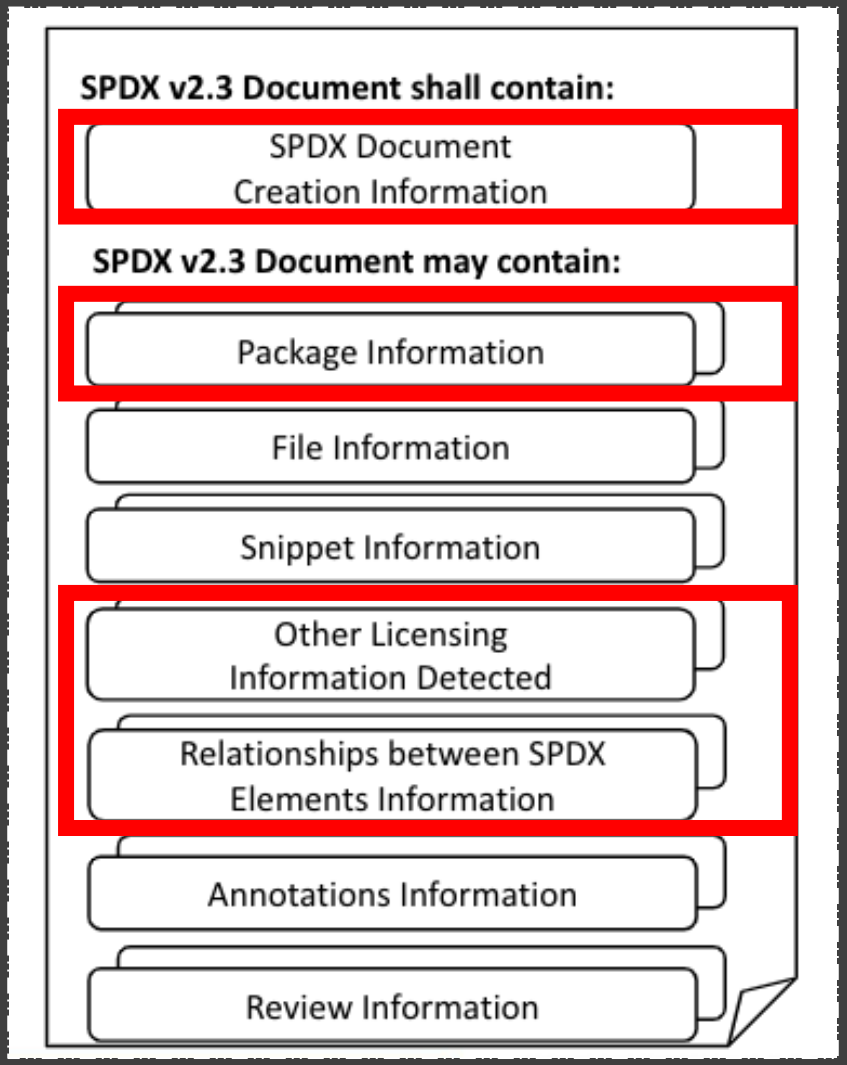


**SBOM ツールに不具合がある、
または利用方法が不明瞭**

- **SBOM ツールが未成熟で、正しい SBOM 生成が難しい**
  - 不具合
  - 機能不足
  - ドキュメント不足や基本的な利用方法などの知見不足
- **エラーの解決法**や**基本的な利用方法**に関する質問が多く存在
  - *“How to create a BOM file in a Flutter project”* [1]
    - Flutter を用いるプロジェクトで CycloneDX 形式の SBOM 生成に失敗

[1] https://stackoverflow.com/questions/76515339 (June 2023)

付録： SPDX 2.3 形式の SBOM の構成 [13]



[13] Linux Foundation, Clause 5: Composition of an SPDX document - specification v2.3.0, 2022

付録： SBOM の種類 [14]

- 「ソフトウェア部品」は多種多様
  - ビルド時に組み込まれて実行時にも使われるもの
  - ビルド時だけ使うもの
  - 実行時に動的に外部から読み込まれるもの
- 米国 CISA の公開文書では SBOM を 6 つに分類

種別	定義
Design	設計段階で作成する SBOM
Source	プロジェクトから作成する SBOM
Build	**ビルド時に作成する SBOM**
Analyzed	**成果物から作成する SBOM**
Deployed	稼働環境全体の SBOM
Runtime	システムの動作を観測して作成する SBOM

[14] Cybersecurity and Infrastructure Security Agency (CISA). “Types of Software Bill of Material (SBOM) Documents,” 2023

付録： 実験環境

- OS: Ubuntu Server 24.04 LTS (AMD64)
- curl 8.10.1 [1]
  - `./configure --with-openssl CFLAGS="-H -Wl,-t"`
    - OpenSSL を使用する
    - 読み込むヘッダファイルとバイナリライブラリを全て出力する
  - `make`
- Apache HTTP Server 2.4.63 [2]
  - `./configure CFLAGS="-H" LDFLAGS="-t"`
    - 読み込むヘッダファイルとバイナリライブラリを全て出力する
  - `make`

[1] https://github.com/curl/curl/releases/download/curl-8_10_1/curl-8.10.1.tar.xz
[2] https://dlcdn.apache.org/httpd/httpd-2.4.63.tar.gz

付録： 評価 – Apache HTTP Server

ビルド時 – 37.2 秒

- 依存ファイル： 390 件
- 依存パッケージ数： 17 件
  - `libssl-dev` と `libssl3t64` を含む

実行ファイル – 4.3 秒

- 依存ファイル： 7 件
- 依存パッケージ数： 7 件

項目	ビルド時	実行時
name *	**100% (17)**	**100% (7)**
versionInfo *	**100% (17)**	**100% (7)**
supplier *	**100% (17)**	**100% (7)**
externalRefs (PURL) *	**100% (17)**	**100% (7)**
packageFileName	100% (17)	100% (7)
downloadLocation	100% (17)	100% (7)
homepage	82.4% (14)	85.7% (6)
summary/description	100% (17)	100% (7)
checksums	100% (17)	100% (7)
copyrightText	29.4% (5)	42.9% (3)
licenseConcluded/Declared	94.1% (16)	100% (7)

付録： 評価 – パッケージメタデータの抽出能力

- SBOM 生成対象： システム上の全パッケージ
  - Web サーバ等の役割を担い実稼働中の Ubuntu 24.04（984 件）
  - VM の Debian 12（1677 件）
  - VM の Linux Mint 22.1（2523 件）

項目	Ubuntu	Debian	Linux Mint
name *	**100% (984)**	**100% (1677)**	**100% (2523)**
versionInfo *	**100% (984)**	**100% (1677)**	**100% (2523)**
supplier *	**100% (984)**	**100% (1677)**	**100% (2523)**
externalRefs (PURL) *	**100% (984)**	**100% (1677)**	**100% (2523)**
packageFileName	95.2% (937)	99.9% (1676)	98.7% (2491)
downloadLocation	95.2% (937)	99.9% (1676)	98.7% (2491)
homepage	81.2% (799)	87.8% (1472)	86.0% (2170)
summary/description	94.5% (930)	99.9% (1676)	90.9% (2294)
checksums	95.2% (937)	99.9% (1676)	98.7% (2491)
copyrightText	70.4% (693)	72.8% (1220)	76.4% (1927)
licenseConcluded/Declared	91.2% (897)	100% (1677)	98.7% (2489)
生成時間	6m25.2s	5m52.5s	21m31.8s

付録： ライセンス情報の取得

- 後述の NTIA 要件には含まれないが、
  Stack Overflow 上の質問内容の調査で需要を確認
  - *"How to include Open Source license in GitHub SBOM export?"* *
- **ライセンス情報はパッケージメタデータに含まれない**
  - `/usr/share/doc/パッケージ名/copyright` を解析する
  - 機械可読形式が提案されており、多くはそれに従う
- **ライセンス表記が SPDX 表現と異なるため修正する**
  - ライセンス名の略記法を変換
    - 大文字小文字やバージョン番号の表記法などの違いを、
      対応表とヒューリスティックで吸収して変換
  - 複数ライセンスの場合の式表現を構文解析し、記法変換
    - `,` `and` `or` による表現を `()` `AND` `OR` による表現に変換
    - SPDX 規格に定義されたライセンス以外は ID を発行し、
      SPDX Document 内の別節で全文を記載

* https://stackoverflow.com/questions/76962834 (August 2023)

付録： 評価 – ライセンス取得能力

- 対象： 本番稼働中サーバの全インストール済みパッケージ
  - OS は Ubuntu Server 24.04 LTS (AMD64)

- **3467 件中 1187 件を特殊ライセンスと判定**
  - SPDX License List に登録されていないライセンス
- **ヒューリスティックを無効化すると 2627 件まで増加**
  - ヒューリスティックは有益に機能している

項目	取得成功率
name	82.0% (973)
extractedText	98.3% (1167)

付録： 今後の課題

- コードベース中に取り込まれた依存関係への対応
  - フォルダごとコードベースにコピーする形で、
    外部ライブラリを導入する場合がある
- 他の OS やコンパイラへの対応
  - RedHat 系 Linux や Windows など
  - Clang や MSVC など
- 機械可読でない copyright ファイルからのライセンス抽出
- 評価対象のソフトウェアを増やす
  - 手法の汎用性や有用性をさらに深く検証