

INFORMATION TECHNOLOGIES

Anhelina Kovalenko

National Aviation University, Kyiv, Ukraine

SOFTWARE BILL OF MATERIALS AS A TOOL FOR SOFTWARE SUPPLY CHAIN RISK MANAGEMENT

Abstract. The paper analyzes the use of a Software Bill of Materials (SBOM) as a formal inventory of components included in a software product. The role of SBOM in dependency discovery, rapid identification of exposure to a vulnerable component, and improved software supply chain transparency is substantiated. An integration process is proposed: automatic generation during build, storage with release artifacts, vulnerability checking, license and component change control, and re-scanning when new vulnerability information appears. An SBOM is not a standalone security control and should be combined with secure development practices and update management.

Keywords: SBOM, software supply chain, cybersecurity, dependencies, vulnerabilities, secure software development.

Ангеліна Коваленко

Національний авіаційний університет, Київ, Україна

ПЕРЕЛІК КОМПОНЕНТІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ (SOFTWARE BILL OF MATERIALS) ЯК ІНСТРУМЕНТ УПРАВЛІННЯ РИЗИКАМИ ЛАНЦЮГА ПОСТАЧАННЯ

Анотація. Проаналізовано використання Software Bill of Materials (SBOM) як формалізованого переліку компонентів, з яких складається програмний продукт. Обґрунтовано роль SBOM у виявленні залежностей, оперативному встановленні факту використання вразливого компонента та підвищенні прозорості ланцюга постачання програмного забезпечення. Запропоновано процес інтеграції SBOM у життєвий цикл розробки: автоматичне формування під час збирання, зберігання разом з артефактами релізу, перевірка на відомі вразливості, контроль ліцензійних та компонентних змін, повторне сканування після появи нової інформації про вразливості. Підкреслено, що SBOM не є самостійним засобом захисту і має використовуватися разом із безпечними практиками розробки та керування оновленнями.

Ключові слова: SBOM, ланцюг постачання програмного забезпечення, кібербезпека, залежності, вразливості, безпечна розробка.

Сучасне програмне забезпечення рідко створюється повністю з власного коду. Навіть невеликий вебсервіс може залежати від десятків або сотень бібліотек, пакетів, контейнерних образів і транзитивних залежностей. Це прискорює розробку, але одночасно створює проблему видимості: команда може не знати, що певний компонент фактично присутній у кінцевому продукті. Коли для такої бібліотеки публікується критична вразливість, перше практичне питання звучить не «як її виправити», а «у яких наших системах вона використовується».

Software Bill of Materials — це формалізований перелік компонентів програмного забезпечення та зв'язків між ними. NTIA визначає SBOM як запис, що містить деталі та відносини ланцюга постачання різних компонентів, використаних для побудови програмного продукту, і встановлює мінімальні елементи такого опису (NTIA, 2021). Ідея є аналогічною до специфікації матеріалів у виробництві: щоб оцінювати ризик продукту, необхідно знати його склад.

Практична цінність SBOM проявляється під час реагування на нову вразливість. За відсутності централізованого переліку команді доводиться опитувати розробників, переглядати репозиторії або запускати термінове сканування великої кількості систем. Якщо ж для кожного релізу зберігається машиночитний SBOM, можна швидко визначити, які продукти містять конкретний пакет і версію. Це не підтверджує автоматично можливість експлуатації, але істотно звужує область перевірки та допомагає пріоритезувати роботу.

Найбільш ефективним є автоматичне формування SBOM у конвеєрі безперервної інтеграції та постачання. Перелік повинен створюватися з фактичного артефакту або середовища збирання, а не заповнюватися вручну. Разом із релізом доцільно зберігати версію SBOM, хеш артефакту та інформацію про середовище, щоб у майбутньому можна було відтворити зв'язок між конкретним файлом і переліком його компонентів. Для контейнеризованих систем важливо враховувати як залежності застосунку, так і компоненти базового образу.

Таблиця 1. Життєвий цикл SBOM у процесі розробки

Етап	Дія	Результат
Збирання	Автоматичне формування SBOM	Фактичний склад релізу
Реліз	Збереження SBOM разом з артефактом	Простежуваність версії
Експлуатація	Повторне зіставлення з даними про вразливості	Виявлення нового ризику без повторного збирання
Оновлення	Порівняння складу та заміна залежностей	Контроль змін і зменшення технічного боргу

Джерело: складено автором на основі NTIA (2021) та Sourprya et al. (2022).

Наступний рівень — постійне зіставлення SBOM із базами відомих вразливостей. Вразливість може бути оприлюднена через тижні або місяці після випуску продукту, тому одноразове сканування перед релізом є недостатнім. Організація має мати можливість повторно перевірити вже випущені версії без їх повторного збирання. При цьому потрібна нормалізація назв пакетів і версій: одна й та сама бібліотека може ідентифікуватися по-різному в різних екосистемах, що створює помилкові збіги або пропуски.

Важливо не ототожнювати SBOM із повною оцінкою безпеки. Наявність у продукті компонента з опублікованою CVE не завжди означає, що вразливий код досяжний у конкретній конфігурації. Так само відсутність відомих CVE не

гарантує відсутності дефектів. SBOM створює прозорість і основу для аналізу, але рішення про ризик потребує контексту: доступності компонента, способу використання, мережевої експозиції, привілеїв і наявності компенсуючих заходів.

Інтеграція SBOM має бути пов'язана із ширшим процесом безпечної розробки. NIST Secure Software Development Framework рекомендує вбудовувати практики безпеки в життєвий цикл розробки, щоб зменшувати кількість вразливостей і їхній потенційний вплив (Souppaya et al., 2022). У цьому контексті SBOM доповнює керування залежностями, перевірку походження коду, контроль змін, захист середовища збирання та своєчасне оновлення компонентів. Якщо організація генерує SBOM, але не має процесу оновлення вразливих залежностей, документ перетворюється на інвентар без механізму реагування.

Для практичного впровадження доцільно визначити мінімальний стандарт: які продукти повинні мати SBOM, на якому етапі він генерується, де зберігається, хто має доступ, як довго підтримується історія та яким чином виконується повторна перевірка. Варто також фіксувати зміни складу між релізами, оскільки додавання нового пакета може змінити не лише функціональність, а й ліцензійні та безпекові ризики. Автоматизований diff двох SBOM може бути частиною процедури погодження релізу.

Отже, SBOM є інструментом прозорості ланцюга постачання програмного забезпечення. Його основна користь полягає у здатності швидко відповісти на питання «що входить до продукту» та «де використовується конкретний компонент». Найкращий ефект досягається тоді, коли SBOM генерується автоматично, зберігається для кожного релізу, регулярно зіставляється з новими даними про вразливості та включений у процес управління оновленнями. У такій моделі перелік компонентів стає не формальним документом, а робочим джерелом даних для безпеки й супроводу ПЗ.

REFERENCES / СПИСОК ДЖЕРЕЛ

1. National Telecommunications and Information Administration. (2021). The Minimum Elements for a Software Bill of Materials (SBOM). U.S. Department of Commerce.
2. Souppaya, M., Scarfone, K., & Dodson, D. (2022). Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities (NIST Special Publication 800-218). National Institute of Standards and Technology. <https://doi.org/10.6028/NIST.SP.800-218>