ISSN 1996-1588

Міністерство освіти і науки України

НАУКОВІ ПРАЦІ

ДОНЕЦЬКОГО НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ

Серія: Інформатика, кібернетика та обчислювальна техніка

Присвячується 105-річчю
ДВНЗ «Донецький національний технічний університет»

№1(42) 2026

1921 ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Дрогобич - 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Наукові праці
Донецького національного технічного університету

Серія: “Інформатика, кібернетика та обчислювальна техніка”

Всеукраїнський науковий збірник

Заснований у травні 1996 року

Виходить 2 рази на рік

№ 1 (42)’ 2026

Дрогобич – 2026

УДК 004+519.6+519.7

Публікується згідно з рішенням Вченої ради ДВНЗ «Донецький національний технічний університет» (протокол № 4 від 16.04.2026).

Збірник містить наукові статті співробітників ДонНТУ та інших навчальних і наукових закладів України, які є науковими партнерами ДонНТУ. Публікації висвітлюють результати наукових досліджень і розробок в таких напрямках, як інформатика, чисельні методи, паралельні обчислення, програмування, розробка засобів обчислювальної техніки, дослідження комп'ютерних мереж, машинна графіка і обробка зображень, математичне моделювання в різних галузях. Матеріали збірника призначені для наукових співробітників, викладачів, інженерно-технічних працівників, аспірантів та студентів.

Засновник та видавець – Донецький національний технічний університет (ДонНТУ)

Адреса редакції:
Юридична адреса: пл. Шибанкова, 2, м. Покровськ, Донецька область, 85300
Фактична адреса: вул. Самбірська, 76, м. Дрогобич, Львівська обл., 82111
Адреса для листування: вул. Шевченка, 9, м. Дрогобич, Львівська обл., 82100
E-mail: yevhen.bashkov@donntu.edu.ua

Збірник зареєстровано в Державному комітеті інформаційної політики, телебачення та радіомовлення України. Свідоцтво: серія КВ, №7374 від 03.06.2003.

Збірник включено до переліку наукових фахових видань України, в яких можуть публікуватися результати дисертаційних робіт на здобуття наукових ступенів доктора наук, кандидата наук та ступеня доктора філософії за спеціальностями 121 Інженерія програмного забезпечення, 122 Комп'ютерні науки, 123 Комп'ютерна інженерія (наказ Міністерства освіти і науки України №409 від 17 березня 2020 р.)

Збірник "Наукові праці ДонНТУ, серія "Інформатика, кібернетика та обчислювальна техніка" за наказом № 409 МОНУ від 17. 03.2020 отримав категорію Б.

ISSN 1996-1588 (Print),
ISSN 3083-6050 (Online)

ЗМІСТ

УДК 004.41:004.056.5:004.052.3

О.О. Дудник, канд. техн. наук, доцент
О.Н. Романюк, д-р. техн. наук, професор
[1,2]Вінницький національный технічний університет, м. Вінниця, Україна
[1]dudnyk@vntu.edu.ua

Software supply chain security: концепції, моделі та виклики сучасності

Систематизовано актуальні концепції, моделі загроз та стандарти у сфері безпеки ланцюга постачання програмного забезпечення (Software Supply Chain Security, SSCS) з позицій інженерії програмного забезпечення. В умовах стрімкого зростання складності програмних систем, широкого використання відкритих компонентів і автоматизованих CI/CD-конвеєрів проблема забезпечення цілісності, автентичності та відстежуваності програмних артефактів набуває критичного значення. У роботі розглянуто структуру ланцюга постачання програмного забезпечення, що охоплює всі етапи життєвого циклу — від розробки вихідного коду до розгортання та супроводу, а також визначено його ключові компоненти, зокрема залежності, інфраструктуру збірки, системи керування версіями та репозиторії артефактів. Проведено аналіз типології вразливостей, характерних для різних етапів життєвого циклу ПЗ, включаючи компрометацію залежностей, атаки на CI/CD-процеси, підміну артефактів і компрометацію механізмів цифрового підпису. Показано, що сучасні атаки дедалі частіше спрямовані не на сам код, а на інфраструктуру його створення та доставки. Окрему увагу приділено провідним міжнародним ініціативам і стандартам, таким як SLSA, SBOM, Sigstore та SSDF, які формують основу для підвищення прозорості, верифікованості та довіри до програмних продуктів. Визначено ключові проблеми впровадження підходів SSCS, зокрема недостатній рівень автоматизації, складність інтеграції у наявні процеси розробки, обмежену якість метаданих та відсутність уніфікованих моделей оцінювання довіри. Окреслено перспективні напрями розвитку, серед яких автоматизована перевірка відповідності стандартам, інтеграція даних про склад ПЗ із базами вразливостей, а також побудова формальних моделей довіри до артефактів на основі криптографічно підтвердженого походження.

Ключові слова: ланцюг постачання ПЗ, безпека програмного забезпечення, SSCS, SLSA, SBOM, Sigstore, залежності, CI/CD, DevSecOps

DOI: 10.31474/1996-1588-2026-1-42-102-108

Вступ

Сучасні методи розробки програмного забезпечення передбачають активне використання сторонніх бібліотек, фреймворків, хмарних сервісів збірки та автоматизовані конвеєри доставки коду. За даними Gartner, до 2025 року понад 45 % організацій зазнають атак на свої ланцюги постачання програмного забезпечення [1]. Така статистика робить проблему безпеки ланцюга постачання ПЗ (Software Supply Chain Security, SSCS) одним із пріоритетних викликів сучасної інженерії програмного забезпечення.

У 2020 році було скомпрометовано платформу SolarWinds Orion [2]. У 2021 році виявлено критичну вразливість Log4Shell у бібліотеці Apache Log4j [3]. У 2024 році виявлено навмисне впровадження бекдору в утиліту XZ Utils [4]. Такі резонансні інциденти останніх років наочно продемонстрували, що традиційні методи забезпечення якості та безпеки ПЗ є недостатніми в умовах складних багатокомпонентних залежностей. Ці події спонукали наукову спільноту та провідні технологічні організації до інтенсивного дослідження відповідних концепцій, моделей та інструментів.

Питання безпеки ланцюгів постачання ПЗ досліджуються у працях зарубіжних науковців, зокрема у роботах Ladisa et al. [5], Ohm et al. [6], а також відображені в офіційних документах NIST [7] та CISA [8]. Водночас вітчизняна наукова література відповідної тематики залишається нечисленною, а систематизованих оглядів із позицій інженерії програмного забезпечення майже немає.

Метою статті є систематизація актуальних концепцій, моделей загроз та стандартів у сфері безпеки ланцюга постачання програмного забезпечення з позицій інженерії ПЗ. Відповідно до мети визначено такі завдання: уточнити структуру

та компоненти ланцюга постачання ПЗ; систематизувати типологію вразливостей у контексті SDLC; проаналізувати провідні міжнародні ініціативи та стандарти; окреслити актуальні проблеми впровадження та перспективні напрями розвитку. Об'єктом дослідження є процеси розробки та постачання програмного забезпечення. Предметом — концепції, моделі та стандарти забезпечення безпеки ланцюга постачання ПЗ.

Поняття та структура ланцюга постачання програмного забезпечення

Ланцюг постачання програмного забезпечення охоплює усі етапи, учасників і артефакти, задіяні у процесі від написання вихідного коду до його розгортання і подальшого оновлення у середовищі виконання [5]. Це поняття є ширшим за традиційне розуміння «розробки ПЗ», оскільки включає зовнішні залежності, інфраструктуру та процеси, що безпосередньо не перебувають під контролем команди розробників (рис. 1).

З інженерної точки зору, ланцюг постачання ПЗ складається з таких основних компонентів:

– вихідний код власної розробки та відкриті бібліотеки (open-source dependencies);

– системи керування версіями (Git, SVN, Mercurial);

– інфраструктура CI/CD (GitHub Actions, GitLab CI, Jenkins, CircleCI);

– системи управління пакетами (npm, PyPI, Maven, NuGet);

– артефакти збірки: контейнерні образи (Docker, OCI), бінарні файли, бібліотеки;

– реєстри та репозиторії артефактів (Docker Hub, Artifact Registry, Nexus);

– процеси розгортання та механізми автоматичного оновлення.

Важливим поняттям у контексті SSCS є поверхня атаки (attack surface) ланцюга постачання. Це сукупність усіх точок, через які зловмисник може впровадити шкідливий код або втрутитися у процес доставки ПЗ [6]. Дослідження Ohm et al. демонструє, що лише 10% проаналізованих атак були спрямовані безпосередньо на вихідний код, тоді як решта були зосереджені на залежності та інфраструктуру збірки [6].

Поняття довіри в контексті SSC визначається через здатність верифікувати походження (provenance) кожного артефакту. Важливо мати можливість чітко встановити хто, де, коли і з яких вхідних даних його створив. Формалізація такої верифікації є центральним завданням сучасних ініціатив безпеки ланцюга постачання ПЗ [7].

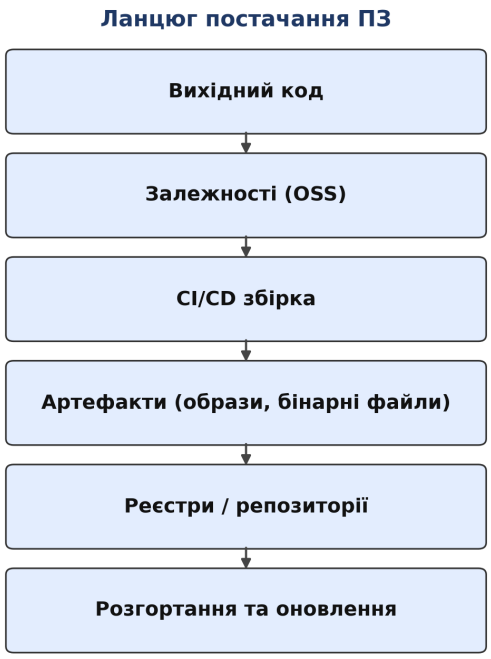

Рисунок 1 – Структура ланцюга постачання програмного забезпечення

Вразливості в SDLC та типологія загроз

Розгляд загроз з позицій інженерії ПЗ передбачає їх прив'язку до конкретних етапів життєвого циклу програмного забезпечення (Software Development Life Cycle, SDLC). Це дозволяє визначити де саме і на якому етапі існує потреба в застосуванні контрзаходів [9].

Ladisa et al. запропонували систематичну таксономію атак на ланцюг постачання ПЗ з відкритим кодом, яка охоплює 107 унікальних векторів атак, організованих у 10 категорій [5]. У контексті SDLC ці вектори можна згрупувати за чотирма основними типами.

Компрометація залежностей. Зараження залежностей або їх підміна. Зловмисник публікує пакет із назвою, що імітує легітимну залежність, або вищою версією у публічному реєстрі. Прикладом атаки типу dependency confusion, описана дослідником A. Birsan у 2021 році [10].

Компрометація CI/CD. Впровадження шкідливого коду в процеси автоматизованої збірки через незахищені змінні середовища, небезпечні GitHub Actions, або зловживання правами доступу до сховища артефактів [11].

Підміна артефактів. Розповсюдження підроблених або модифікованих пакетів, образів контейнерів чи бінарних файлів через скомпрометовані реєстри або атаки типу man-in-the-middle під час завантаження [5].

Атаки на ключі цифрових підписів. Компрометація облікових даних для підписання коду, що дозволяє зловмиснику видавати шкідливі артефакти за легітимні, підписані довіреним розробником [12].

Для структурованого опису поведінки зловмисників у контексті SSC застосовується фреймворк MITRE ATT&CK for Supply Chain. Він визначає тактики, техніки та процедури (TTP), характерні для атак на ланцюги постачання, і є стандартним інструментом моделювання загроз в індустрії [13].

Показовим прикладом комплексної атаки є інцидент SolarWinds 2020 року: зловмисники (група Nobelium/APT29) впровадили шкідливий код безпосередньо в процес збірки платформи Orion, в результаті чого скомпрометоване оновлення було доставлено приблизно 18000 організацій-клієнтів [2]. Цей інцидент увійшов до наукової літератури як еталонний приклад атаки типу build-time compromise [9].

Основні ініціативи та стандарти

Реакцією індустрії та наукової спільноти на зростання загроз SSC стало формування розгалуженої екосистеми стандартів, фреймворків та інструментів. Розглянемо ключові з них.

Модель SLSA. SLSA (*Supply-chain Levels for Software Artifacts*) – фреймворк забезпечення цілісності ланцюга постачання ПЗ, розроблений Google та переданий під управління OpenSSF [14]. Модель визначає чотири рівні зрілості (SLSA 1-4). Кожен з них встановлює дедалі суворіші вимоги до процесу збірки та верифікації артефактів (рис. 2).

Рисунок 2 – Рівні моделі зрілості SLSA та їх ключові вимоги

Центральним поняттям SLSA є provenance – криптографічно підписаний метадата-документ, що описує як, де і з яких джерел було зібрано артефакт. Починаючи з рівня SLSA 2, provenance має бути підписано сервісом збірки, а не самим розробником, що унеможливлює його фальсифікацію.

SBOM – специфікація складу програмного забезпечення. Software Bill of Materials (SBOM) – структурований перелік усіх компонентів, бібліотек та залежностей програмного продукту із зазначенням їх версій, ліцензій та відомих вразливостей [15].

Концепція SBOM набула регуляторного статусу в США після Указу президента США № 14028 (2021), який зобов'язав постачальників ПЗ для федеральних органів надавати SBOM [8]. Наразі існують два домінуючих формати SBOM: *SPDX* (стандарт ISO/IEC 5962:2021) та *CycloneDX* (стандарт ECMA-424). Вони відрізняються семантичною моделлю та сферою застосування. SPDX орієнтований на ліцензійний комплаєнс (рис. 3). CycloneDX орієнтований переважно на безпекові сценарії використання [15].

Рисунок 3 – Екосистема інструментів безпеки ланцюга постачання ПЗ

Sigstore та підписання артефактів**.** Sigstore є відкритою екосистемою для безпечного підписання та верифікації програмних артефактів без потреби управляти довгостроковими ключами підписання [16]. Вона включає три основних компоненти: Cosign (підписання контейнерів та файлів), Fulcio (центр сертифікації, що видає

короткострокові сертифікати), та Rekor (незмінний публічний лог підписів).

OpenSSF та SSDF. Open Source Security Foundation (OpenSSF) координує розробку низки інструментів та стандартів, зокрема OpenSSF Scorecard.

Це набір автоматизованих перевірок безпеки для open-source проєктів за 18 категоріями [17].

NIST Secure Software Development Framework (SSDF, SP 800-218) визначає набір рекомендованих практик безпечної розробки ПЗ, структурованих навколо чотирьох груп: підготовка організації, захист ПЗ, розробка захищеного ПЗ та реагування на вразливості [7].

SSDF став основою для формування вимог до постачальників ПЗ для федерального уряду США.

Поточний стан і проблеми впровадження

Попри активний розвиток стандартів та інструментів, практичне впровадження SSCS стикається з низкою системних проблем, що фіксуються як у науковій літературі, так і у галузевих звітах.

Якість SBOM. Дослідження Balliu et al. виявило, що автоматично згенеровані SBOM часто містять неповні, дубльовані або суперечливі записи [18]. Проблема загострюється при роботі з транзитивними залежностями — пакетами, що підключаються непрямо через інші залежності. За даними звіту Sonatype (2023), середній застосунок містить 49 транзитивних залежностей на кожну пряму [19].

Відсутність єдиних метрик довіри. На сьогодні не існує стандартизованого механізму агрегації даних з різних джерел (SBOM, provenance, підписи, дані вразливостей) у єдину оцінку довіри до артефакту. OpenSSF Scorecard вирішує частину цієї проблеми, однак охоплює лише відкриті проєкти на GitHub [17].

Обмежена автоматизація SLSA-аудиту. Реальні CI/CD-конвеєри є складними та різнорідними системами. Тому автоматична перевірка їх відповідності вимогам SLSA 3–4 залишається відкритою дослідницькою проблемою. Більшість організацій, за даними опитування CNCF (2023), не досягли навіть рівня SLSA 2 [20].

Несумісність форматів. Паралельне існування форматів SPDX і CycloneDX ускладнює інтеграцію інструментів у гетерогенних середовищах. Незважаючи на наявність конверторів, семантична еквівалентність між форматами не є повною [15].

Проблема зберігання ключів та довіри до логів. Хоча Sigstore/Rekor надає незмінний публічний лог підписів, довгострокова верифікація підписів (наприклад, через 5–10 років після виходу версії) ставить питання щодо стійкості криптографічних алгоритмів та доступності лог-сервісів [16].

Крім того, важливо звернути увагу на кадровий аспект. Більшість розробників не мають достатніх знань у сфері SSCS. Дослідження Zahan et al. показало, що лише 17% мейнтейнерів відкритих проєктів усвідомлюють ризики, пов'язані із залежностями [21].

Моделі довіри та перспективи розвитку

Актуальним завданням інженерії ПЗ є побудова формальних моделей довіри до артефактів (artifact trust models), що інтегрують гетерогенні дані з різних джерел (SBOM, provenance-звітів, підписів, журналів збірки, відомостей про вразливості) у структуровану оцінку ризику для конкретної версії компонента.

Torres-Arias et al. запропонували фреймворк in-toto, що дозволяє формально специфікувати та верифікувати ланцюжок дій у конвеєрі розробки через метадата-файли link та layout [12]. Цей підхід забезпечує криптографічно верифіковане свідоцтво того, що кожен крок SDLC виконувався в очікуваному порядку очікуваними учасниками.

Серед перспективних напрямів досліджень і розробок варто виділити такі такі:

– автоматизована перевірка SLSA-відповідності у гетерогенних CI/CD-середовищах, включаючи гібридні та мультихмарні конфігурації;

– розробка стандартизованих метрик якості SBOM та методів їх автоматичної верифікації;

– інтеграція SBOM з базами даних вразливостей (OSV.dev, NVD) у реальному часі для автоматичного оцінювання ризику компонентів;

– побудова графів довіри між артефактами для виявлення транзитивних ризиків у великих екосистемах залежностей;

– застосування методів статичного аналізу та машинного навчання для виявлення шкідливих пакетів у реєстрах до їх завантаження;

– інтеграція практик SSCS в освітні програми з інженерії програмного забезпечення як обов'язкового компонента підготовки фахівців.

Особливої уваги заслуговує питання регуляторного контексту. У США Указ 14028 та подальші настанови CISA сформували чіткі вимоги для постачальників ПЗ [8].

У Європейському Союзі Акт про кіберстійкість (Cyber Resilience Act, 2024) запроваджує обов'язкові вимоги до безпеки ПЗ для виробників цифрових продуктів. В Україні питання регулювання SSCS поки що не відображено у профільному законодавстві, що є прогалиною, яку необхідно усунути в контексті розвитку вітчизняної IT-індустрії.

Висновки

Безпека ланцюга постачання програмного забезпечення сформувалась як самостійна дисципліна інженерії ПЗ. Вона відповідає на системну вразливість сучасних програмних систем: складність і непрозорість їхніх компонентних залежностей. Аналіз поточного стану дозволяє сформулювати такі висновки.

Ланцюг постачання ПЗ є складною соціотехнічною системою, кожен елемент якої (від вихідного коду до механізму оновлення) є потенційною точкою компрометації. Ефективний захист вимагає системного підходу, що охоплює всі етапи SDLC.

Сформована екосистема стандартів (SLSA, SBOM, Sigstore, SSDF) закладає формальну основу для верифікації цілісності та походження артефактів. Однак ефективне впровадження цих стандартів стримується проблемами автоматизації, якості метаданих та несумісності форматів. Перспективним напрямом є розробка інтегрованих моделей довіри до артефактів, що агрегують дані з різних джерел, та їх вбудовування в стандартні CI/CD-конвеєри як невід'ємної складової процесу доставки ПЗ. Безпека ланцюга постачання ПЗ повинна стати обов'язковою компетентністю інженерів-програмістів, що вимагає відповідної інтеграції у навчальні програми вищої освіти та програми підвищення кваліфікації фахівців

Література

1. Gartner predicts 45% of organizations worldwide will have experienced attacks on their software supply chains by 2025. Gartner Research, 2022.
2. Peisert S., Schneier B., Okhravi H. et al. Perspectives on the SolarWinds incident. *IEEE Security & Privacy*. 2021. Vol. 19, № 2. P. 7–13. DOI:10.1109/MSEC.2021.3051235.
3. Chen J., Kelkar S., Mazurek M. L. A study of Log4Shell: The Log4j vulnerability. *Proceedings of the 2022 IEEE/ACM International Conference on Mining Software Repositories (MSR)*. 2022. P. 300–304.
4. Freund E. Backdoor in upstream xz/liblzma leading to SSH server compromise. 2024. URL: https://www.openwall.com/lists/oss-security/2024/03/29/4 (дата звернення: 20.03.2026).
5. Ladisa G., Plate H., Martinez M., Barais O. A taxonomy of attacks on open-source software supply chains. *Proceedings of the 44th IEEE Symposium on Security and Privacy (S&P)*. 2023. P. 1509–1526. DOI: 10.1109/SP46215.2023.10179304.
6. Ohm M., Plate H., Sykosch A., Meier M. Backstabber's knife collection: A review of open source software supply chain attacks. *Proceedings of the 17th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*. 2020. P. 1–20. DOI: 10.1007/978-3-030-52683-2_2.
7. Dodson D., Souppaya M., Scarfone K. NIST special publication 800-218: Secure software development framework (SSDF), version 1.1. Gaithersburg: NIST, 2022. 36 p.
8. CISA. Software supply chain security guidance under Executive Order (EO) 14028 section 4e. 2023. URL: https://www.cisa.gov/resources-tools/resources/software-supply-chain-security-guidance (дата звернення: 20.03.2026).
9. Enck W., Williams L. Top five challenges in software supply chain security: Observations from 30 industry and government organizations. *IEEE Security & Privacy*. 2022. Vol. 20, № 2. P. 96–100. DOI:10.1109/MSEC.2022.3142338.
10. Birsan A. Dependency confusion: How I hacked into Apple, Microsoft and dozens of other companies. *Medium*. 2021. URL: https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610 (дата звернення: 20.03.2026).
11. Koishybayev I., Nahapetyan A., Zachariah R. et al. Characterizing the security of GitHub CI workflows. *Proceedings of the 31st USENIX Security Symposium*. 2022. P. 2747–2763.
12. Torres-Arias S., Afzali H., Bhatt T. K. et al. in-toto: Providing farm-to-table guarantees for bits and bytes. *Proceedings of the 28th USENIX Security Symposium*. 2019. P. 1393–1410.
13. MITRE. ATT&CK for enterprise: Supply chain compromise (T1195). 2023. URL: https://attack.mitre.org/techniques/T1195/ (дата звернення: 20.03.2026).
14. Dolan-Gavitt B. SLSA: Supply-chain levels for software artifacts. *ACM Queue*. 2022. Vol. 20, № 3. P. 30–54.
15. Stalnaker T., Sahinoglu M., Nakamura S. Software bill of materials: Concepts and practices. *IEEE Access*. 2023. Vol. 11. P. 34521–34535.
16. Newman L. E. Sigstore: A solution to software supply chain security. *;login: (USENIX)*. 2022. Vol. 47, № 2. P. 11–16.
17. Zahan N., Zimmermann T., Bhatt P., Williams L. OpenSSF scorecard: Automating security assessment for open source projects. *Proceedings of the 44th International Conference on Software Engineering (ICSE)*. 2022. P. 1–12.

18. Balliu M., Cassel B., Gebremedhin A., Söderberg R. Challenges in generating and validating software bill of materials. *IEEE Transactions on Software Engineering*. 2023. Vol. 49, № 8. P. 4021–4037.
19. Sonatype. 9th annual state of the software supply chain report. Sonatype, 2023. 87 p.
20. CNCF. CNCF annual survey 2023: Security practices in cloud native environments. Cloud Native Computing Foundation, 2023. 42 p.
21. Zahan N., Rashid R., Bhatt P. et al. What are weak links in the npm supply chain? *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. 2022. P. 331–340. DOI: 10.1145/3510457.3513044.

References

1. Gartner (2022), "Gartner predicts 45% of organizations worldwide will have experienced attacks on their software supply chains by 2025", *Gartner Research*, 2022.
2. Peisert, S., Schneier, B., Okhravi, H. et al. (2021), "Perspectives on the SolarWinds incident", *IEEE Security & Privacy*, Vol. 19, No. 2, pp. 7–13. DOI: 10.1109/MSEC.2021.3051235.
3. Chen, J., Kelkar, S., Mazurek, M. L. (2022), "A study of Log4Shell: The Log4j vulnerability", *Proceedings of the 2022 IEEE/ACM International Conference on Mining Software Repositories (MSR)*, pp. 300–304.
4. Freund, E. (2024), "Backdoor in upstream xz/liblzma leading to SSH server compromise", available at: https://www.openwall.com/lists/oss-security/2024/03/29/4.
5. Ladisa, G., Plate, H., Martinez, M., Barais, O. (2023), "A taxonomy of attacks on open-source software supply chains", *Proceedings of the 44th IEEE Symposium on Security and Privacy (S&P)*, pp. 1509–1526. DOI: 10.1109/SP46215.2023.10179304.
6. Ohm, M., Plate, H., Sykosch, A., Meier, M. (2020), "Backstabber's knife collection: A review of open source software supply chain attacks", *Proceedings of the 17th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, pp. 1–20. DOI: 10.1007/978-3-030-52683-2_2.
7. Dodson, D., Souppaya, M., Scarfone, K. (2022), *NIST special publication 800-218: Secure software development framework (SSDF), version 1.1*, NIST, Gaithersburg, 36 p.
8. CISA (2023), "Software supply chain security guidance under Executive Order (EO) 14028 section 4e", available at: https://www.cisa.gov/resources-tools/resources/software-supply-chain-security-guidance.
9. Enck, W., Williams, L. (2022), "Top five challenges in software supply chain security: Observations from 30 industry and government organizations", *IEEE Security & Privacy*, Vol. 20, No. 2, pp. 96–100. DOI: 10.1109/MSEC.2022.3142338.
10. Birsan, A. (2021), "Dependency confusion: How I hacked into Apple, Microsoft and dozens of other companies", *Medium*, available at: https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610.
11. Koishybayev, I., Nahapetyan, A., Zachariah, R. et al. (2022), "Characterizing the security of GitHub CI workflows", *Proceedings of the 31st USENIX Security Symposium*, pp. 2747–2763.
12. Torres-Arias, S., Afzali, H., Bhatt, T. K. et al. (2019), "in-toto: Providing farm-to-table guarantees for bits and bytes", *Proceedings of the 28th USENIX Security Symposium*, pp. 1393–1410.
13. MITRE (2023), "ATT&CK for enterprise: Supply chain compromise (T1195)", available at: https://attack.mitre.org/techniques/T1195/.
14. Dolan-Gavitt, B. (2022), "SLSA: Supply-chain levels for software artifacts", *ACM Queue*, Vol. 20, No. 3, pp. 30–54.
15. Stalnaker, T., Sahinoglu, M., Nakamura, S. (2023), "Software bill of materials: Concepts and practices", *IEEE Access*, Vol. 11, pp. 34521–34535.
16. Newman, L. E. (2022), "Sigstore: A solution to software supply chain security", *login: (USENIX)*, Vol. 47, No. 2, pp. 11–16.
17. Zahan, N., Zimmermann, T., Bhatt, P., Williams, L. (2022), "OpenSSF scorecard: Automating security assessment for open source projects", *Proceedings of the 44th International Conference on Software Engineering (ICSE)*, pp. 1–12.
18. Balliu, M., Cassel, B., Gebremedhin, A., Söderberg, R. (2023), "Challenges in generating and validating software bill of materials", *IEEE Transactions on Software Engineering*, Vol. 49, No. 8, pp. 4021–4037.
19. Sonatype (2023), *9th annual state of the software supply chain report*, Sonatype, 87 p.
20. CNCF (2023), *CNCF annual survey 2023: Security practices in cloud native environments*, Cloud Native Computing Foundation, 42 p.
21. Zahan, N., Rashid, R., Bhatt, P. et al. (2022), "What are weak links in the npm supply chain?", *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pp. 331–340. DOI: 10.1145/3510457.3513044.