УДК 004.056.5
М. Стебельський; Н. Загородна, к. т. н., доц.
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ПРОБЛЕМАТИКА ІНТЕГРАЦІЇ ЗАСОБІВ SCA У ПРОЦЕСИ DEVSECOPS ДЛЯ NODE.JS ПРОЕКТІВ

UDC 004.056.5
M. Stebelskyi; N. Zagorodna, PhD., Assoc. Prof.

PROBLEMATICS OF INTEGRATING SCA TOOLS INTO DEVSECOPS PROCESSES FOR NODE.JS PROJECTS

У сфері сучасної веб-розробки широкого розповсюдження набула платформа Node.js. Вона являє собою середовище виконання JavaScript, що дозволяє створювати високопродуктивні та масштабовані мережеві системи. Невід'ємною складовою цієї інфраструктури виступає пакетний менеджер npm, який забезпечує доступ до найбільшого у світі реєстру бібліотек відкритого коду. Проте широке використання сторонніх модулів та глибока вкладеність залежностей роблять проекти критично вразливими до атак на ланцюжок постачання (Supply Chain Attacks). Оскільки значна частина коду базується на open-source компонентах, забезпечення автоматизованого контролю їх безпеки є пріоритетним завданням, адже ручний аудит у таких масштабах стає неможливим [1].

Проблематика багатьох існуючих інструментів SCA (Software Composition Analysis) полягає у фокусуванні на кількісних показниках та високому рівні «шуму». Більшість стандартних рішень здійснюють поверхневий огляд, не перевіряючи, чи дійсно вразливий компонент використовується у робочому коді програми. Це призводить до блокування релізів через бібліотеки, які фактично не становлять загрози в конкретному проекті. Крім того, типові сканери реагують лише на вже задокументовані в офіційних базах вразливості, залишаючи поза увагою приховані загрози, такі як автоматичний запуск шкідливих скриптів безпосередньо під час встановлення пакетів. Відсутність розумної пріоритезації ризиків ускладнює роботу в швидких циклах розробки, що призводить до явища «втоми від сповіщень», коли розробники ігнорують звіти безпеки заради збереження темпів роботи [2].

Для побудови надійного захисту необхідний перехід від формальної наявності сканера до аналізу його реальної ефективності в умовах CI/CD. Ключовими критеріями стають не лише повнота бази вразливостей, а й точність побудови дерева залежностей, швидкість роботи та здатність мінімізувати помилкові тривоги. Такий підхід дозволяє перетворити моніторинг із блокуючого фактора на інструмент превентивного виявлення загроз [3].

Проведення комплексного порівняльного аналізу функціональності сучасних open-source та пропрієтарних рішень є необхідним етапом для побудови безпечної архітектури. Розробка методики вибору та інтеграції інструментів моніторингу, що базується на об'єктивних метриках точності та продуктивності, дозволяє суттєво знизити ризики компрометації Node.js-додатків та оптимізувати процеси DevSecOps.

Література

1. Supply chain attacks and NPM security [Електронний ресурс] // MDN Web Docs. – 2024. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/Security/Attacks/Supply_chain_attacks.
2. Defending Against Software Supply Chain Attacks [Електронний ресурс] // Cybersecurity and Infrastructure Security Agency (CISA) & NIST. – 2021. – Режим доступу: https://www.cisa.gov/sites/default/files/publications/defending_against_software_supply_chain_attacks_508.pdf.
3. Ensor M. Shifting left on security: Securing software supply chains [Електронний ресурс] / M. Ensor, D. Stevens // Google Cloud Whitepaper. – 2021. – Режим доступу: https://cloud.google.com/files/shifting-left-on-security.pdf.