УДК 004.056.55

А. В. Приймак

АНАЛІЗ МОЖЛИВОСТЕЙ ЗАХИСТУ ДОДАТКІВ ВІД ШКІДЛИВОГО КОДУ ОТРИМАНОГО З МЕНЕДЖЕРА ПАКУНКІВ NPM

Вінницький національний технічний університет

Анотація

В роботі розглянуто проблему безпеки коду сторонніх бібліотек з репозиторію npm. Згідно проведеного аналізу було визначено і запропоновано перелік можливостей для захисту додатків від потенційно вразливих пакунків.

Ключові слова: додаток, npm, Node.js, шкідливий код.

Abstract

The paper considers the problem of security of third-party library code from the npm repository. According to the analysis, a list of options for protecting applications from potentially vulnerable packages was identified and proposed.

Keywords: application, npm, node.js, malicious code.

Вступ

В наш час питання захисту інформації стало невід'ємною частиною будь якого сервісу, додатку чи платформи [1]. Забезпечення захисту додатків починається ще на етапі їх розробки, тому необхідно відповідально відноситись до вибору мови програмування чи технології розробки. Однією з найпопулярніших таких технологій на сьогодні є Node.js. Можливість створювати кросплатформені програми, за допомогою бібліотек розроблених іншими розробниками значно спрощує і пришвидшує сам процес розробки. Таким репозиторієм сторонніх бібліотек в Node.js являється npm [2] (Node Package Manager). При всіх перевагах використання npm, також йому притаманні недоліки, такі як безпека коду сторонніх бібліотек. Згідно з недавнім опитуванням безпеки, проведеним npm, 77% респондентів були стурбовані безпекою OSS/стороннього коду. Вчені виявили безліч вразливих місць в модулях Node.js, які, роблять програми вразливими, тому є актуальним аналіз можливостей захисту додатків від шкідливого коду отриманого з менеджера пакунків npm.

Результати дослідження

У листопаді 2018 року зловмисний пакет під назвою «flatmap-stream» [3] був виявлений як транзитивна залежність популярної бібліотеки «event-stream», який в середньому завантажується 1,4 мільйони разів щотижня. В цьому випадку зловмисник отримав права на публікацію завдяки соціальній інженерії, націлившись на пакет, який регулярно не підтримувався. Зловмисник опублікував оновлену версію «3.3.6», додавши шкідливий код для викрадення криптовалюти. Це не виявлялося протягом двох-трьох місяців.

В іншому інциденті в червні 2019 року було виявлено зловмисний пакет «electron-native-notify» [4], що викрадає конфіденційну інформацію, таку як паролі до гаманців криптовалюти та інші дані. Зловмисник чекав, поки пакет буде використаний іншою популярною бібліотекою, перш ніж вводити шкідливий код у наступні випуски. Це також не було виявлено протягом двох-трьох місяців [5].

Проведений аналіз показав, що існує декілька речей, які можна зробити, щоб захиститися від зловмисних бібліотек. Перше, що потрібно зробити, це усвідомлювати кількість модулів, встановлених у програмі. Ви завжди повинні знати, від скількох модулів залежить ваша програма. Якщо ви коли-небудь знайдете два модулі, що забезпечують однакову функціональність, віддайте перевагу тому, який має менше залежностей. Мати менше залежностей означає мати меншу поверхню атаки.

Великі компанії можуть використовувати груповий аудит кожного пакета та генерувати білий список версій пакетів, які дозволено використовувати розробникам цих компаній. Хоча такий підхід є не дуже практичним, якщо взяти до уваги величезну кількість пакетів, доступних на npm, і кількість їх версій, які оновлюються. Крім того, багато розробників пакетів регулярно випускають оновлення

безпеки, щоб споживачі отримували автоматичне оновлення, але якщо процес перевірки є повільним, то програма може виявитися менш безпечною.

Також в npm існує інструмент NSP, який проводить перевірку усіх модулей і порівнює з чорним списком модулів/версій, які містять відомі вразливості. Запуск npm install навіть повідомить, чи є відомі вразливості. Запуск виправлення аудиту npm спробує замінити вразливі пакети на сумісні з версіями, якщо вони існують. Але варто зазначити, що з 2018 року цей інструмент є скомпрометованим і більше не є рекомендованим до використання. Замість ного npm використовує вбудований інструмент із схожим функціоналом для обробки тих же задач.

Часто проблеми виявляються під час перевірки npm для пакетів, які ще не мають виправленої версії (наприклад, із пакетом stringstream). Наприклад, якщо пакет А не часто оновлюється і це залежить від вразливої версії пакету В, а потім супровідник пакету В випускає виправлення, власник програми не може просто оновити версію пакета В, на яку покладається пакет А. Ще одним недоліком є те, що іноді результати аудиту стосуються проблем, які неможливо використати, наприклад, уразливість ReDoS в модулі, який ніколи не отримує рядки від кінцевих користувачів.

Іншою можливістю захистити власний додаток є відмова від сторонніх бібліотек. Але це є абсолютно недоцільним, оскільки далеко не всі модулі в npm є вразливими, а відтворення стороннього коду власними силами буде надзвичайно ресурсозатратним процесом. Привабливість побудови програм на Node.js походить саме від великої екосистеми модулів на npm, і як результат швидкості, з якою розробники можуть створювати готові до релізу додатки.

Висновки

Отже, в ході проведеного аналізу, стало зрозуміло, що завжди необхідно пам’ятати про встановлені модулі, пам’ятати про дерево залежностей, обережно ставитись до модулів із великою кількістю залежностей та ретельно вивчати модулі, які плануються до встановлення. Це найкращі речі, які можна зробити, щоб запобігти потраплянню шкідливих модулів у дерево залежностей. Потрібно завжди оновлювати модулі, коли вони потрапляють у залежність, оскільки це хороший спосіб отримати виправлення безпеки.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Нікіфорова Л. Моделювання вибору оптимального методу протидії загрозам інформаційної безпеки / Л. Нікіфорова, Ю. Яремчук, А. Шиян. // Реєстрація, зберігання і обробка даних. – 2014. – С. 28–33.

2. npm [Electronic Resource]. – Mode of access : URL : https://uk.wikipedia.org/wiki/Npm. - Назва з екрану.

3. Small world with high risks: A study of security threats in the npm ecosystem / M. Zimmermann, C. Staicu, C. Tenny, M. Pradel. // 28th Security Symposium. – 2019. – С. 995–1010.

4. Investigating The Reproducibility of NPM Packages / [P. Goswami, S. Gupta, Z. Li та ін.]. // 2020 IEEE International Conference on Software Maintenance and Evolution (ICSME). – 2020. – С. 677–681.

5. Towards Using Source Code Repositories to Identify Software Supply Chain Attacks / [D. Vu, I. Pashchenko, F. Massacci та ін.]. // In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. – 2020. – С. 2093–2095.

Приймак Андрій Васильович — аспірант, факультет менеджменту та інформаційної безпеки, Вінницький національний технічний університет, Вінниця, e-mail: andrii.pryimak@live.com.

Pryimak Andrii Vasyliovych — postgraduate, faculty of Management and Information Security, Vinnitsa National Technical University, Vinnitsa, e-mail: andrii.pryimak@live.com.