

УДК 004.056.5

Liliia Abramenko¹, Nataliia Zhukova²

¹ PhD student NU “Zaporizhzhia Polytechnic”

² PhD (Philology), Associate Professor NU “Zaporizhzhia Polytechnic”

THE IMPACT OF JAVASCRIPT ON SERVER-SIDE CYBERSECURITY: ANALYSIS OF NODE.JS VULNERABILITIES IN THE CONTEXT OF EMERGING ATTACKS

This report will discuss the growing cybersecurity challenges facing JavaScript and Node.js applications, focusing on vulnerabilities reported in the Common Vulnerabilities and Exposures (CVE) database for 2022-2024.

General trends in cybersecurity for web technologies are as follows: JavaScript, being the backbone of modern web applications, is increasingly targeted by attackers due to its widespread use in both client-side and server-side environments, such as Node.js [1; 5]. It is essential to highlight that vulnerabilities in JavaScript ecosystems, including XSS (Cross-Site Scripting), command injection, and server-side JavaScript injection (SSJI), have been a significant concern, especially as the number of registered CVE entries continues to rise [1; 2; 5].

Based on available data on the estimated number of CVE reports related to JavaScript vulnerabilities, the total number of CVE entries grew from approximately 25,000 in 2022 to 28,817 in 2023 and reached 40,003 in 2024. If we assume that

JavaScript-related vulnerabilities (including XSS, JS engine issues, and library flaws) account for 5–10% of these, we can estimate the following:

- 2022: 1,250–2,500 JavaScript-related vulnerabilities.
- 2023: 1,441–2,882 JavaScript-related vulnerabilities.
- 2024: 2,000–4,000 JavaScript-related vulnerabilities.

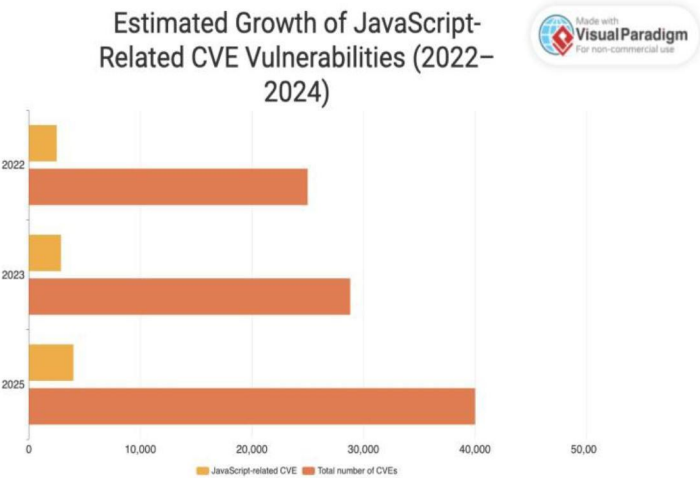


Figure 1. The estimated growth of JavaScript-related CVE vulnerabilities from 2022 to 2024, assuming the upper estimate is 10% from all vulnerabilities

What is particularly interesting here is the upward trend in these numbers, reflecting the growing complexity of JavaScript ecosystems and the increasing reliance on third-party libraries, especially in Node.js applications, which are prone to issues like command injection and SSJI [1; 2; 4]. Figure 1 shows that the number of vulnerabilities has increased by approximately 60% from 2022 to 2024, highlighting the growing attack surface in JavaScript-based platforms.

This brings us to the following key point: the vulnerabilities affecting JavaScript and Node.js applications. It is essential to highlight that server-side JavaScript injection (SSJI) is a critical vulnerability, where attackers inject malicious JavaScript code into server-side logic, potentially leading to data breaches or remote code execution. Additionally, command injection vulnerabilities in Node.js, often stemming from improper input sanitization, allow attackers to execute arbitrary system commands [1; 4].

Some real-world examples and their implications are as follows. Vulnerabilities in Node.js, such as those related to HTTP/2 Rapid Reset (CVE-2023-44487), exposed systems to denial-of-service (DoS) attacks [7], while

flaws in third-party libraries, like those used in Node.js, have been exploited for typosquatting attacks, compromising the software supply chain [6].

Reliance on external dependencies in JavaScript ecosystems, particularly in Node.js, significantly increases the risk of attacks [1; 4; 6]. These findings are significant because they underscore the need for robust security practices, such as those outlined in Node.js security guidelines, which recommend strict input validation, dependency auditing, and the use of security headers [3; 4].

In conclusion, this study has examined the escalating prevalence of JavaScript-related vulnerabilities, estimated at 4,700–9,400 CVE entries over 2022–2024, the upward trend in attacks, and the specific risks posed by vulnerabilities like SSJI and command injection in Node.js applications. Given these challenges, securing JavaScript and Node.js applications requires a proactive approach, including regular updates, vulnerability scanning, and adherence to secure coding practices [3–5]. The increasing number of vulnerabilities underscores the urgency of these measures in the face of evolving threats.

REFERENCES

1. Tal L. JavaScript Security Issues in Node.js Applications [Electronic resource] / L. Tal. – Access mode: <https://www.nodejs-security.com/blog/javascript-security-issues-nodejs-applications/> (accessed: 2025.03.18). – Title from the monitor.
2. A Deep Dive into Server-Side JavaScript Injection (SSJI) Vulnerabilities [Electronic resource]. – 2023. – Access mode: <https://secops.group/a-deep-dive-into-server-side-javascript-injection-ssji-vulnerabilities/> (Accessed: 2025.03.18). – Title from the monitor.
3. Gonzaga R. Security Best Practices [Electronic resource] / R. Gonzaga. – Access mode: <https://nodejs.org/en/learn/getting-started/security-best-practices/> (Accessed: 2025.03.18). – Title from the monitor.
4. Ulili S. Best Practices for Securing Node.js Applications in Production [Electronic resource] / S. Ulili. – Access mode: <https://betterstack.com/community/guides/scaling-nodejs/securing-nodejs-applications/> (Accessed: 2025.03.18). – Title from the monitor.
5. Top 9 Javascript security vulnerabilities in 2025 [Electronic resource]. – Access mode: <https://zcybersecurity.com/javascript-security-vulnerabilities/> (accessed: 2025.03.18). – Title from the monitor.
6. Critical Vulnerabilities in Node.js Expose Systems to Remote Attacks [Electronic resource]. – Access mode: <https://cyble.com/blog/critical-vulnerabilities-in-node-js-expose-systems/> (accessed: 2025.03.18). – Title from the monitor.

7. CVE-2023-44487 [Electronic resource]. – Access mode:
<https://www.cve.org/CVERecord?id=CVE-2023-44487/> (accessed:
2025.03.18). – Title from the monitor.