Trade-Offs Between Efficiency and Security in AI-Assisted Software Development

Muhammed Ali Elhasan, Yousef Abbas, Zakaria Aslan
Department of Computer Science, University of Gothenburg

***Abstract*—Generative AI is transforming software development by enhancing productivity through automated coding assistance. However, while AI-assisted tools such as GitHub Copilot accelerate development, they introduce security vulnerabilities that may compromise software integrity. This study employs a mixed-methods approach, combining surveys of developers and code analysis, to investigate the trade-offs between efficiency and security risks. The findings aim to provide actionable insights for balancing rapid development with secure coding practices, benefiting both industry practitioners and researchers.**

***Index Terms*—Generative AI, Software Security, Productivity, AI-Assisted Development.**

I. Introduction

The adoption of AI-assisted programming tools, such as GitHub Copilot and ChatGPT-based code generation, has reshaped software development workflows. These tools promise increased developer productivity [1], [2] while raising concerns about security vulnerabilities in AI-generated code [3], [4].

Studies indicate that AI-assisted developers complete coding tasks significantly faster than those without AI assistance [1]. IBM's internal study of AI pair programming tools found that while most developers reported efficiency gains, factors such as task type and experience level influenced the actual benefits [2]. However, industry reports challenge these findings, with Uplevel Data Labs showing that Copilot users introduced more bugs without an overall increase in productivity [5].

Security evaluations have revealed that a substantial portion of AI-generated code contains vulnerabilities [3], [4]. Perry et al. found that AI-assisted developers unknowingly wrote insecure code while maintaining high confidence in its correctness [4]. Similar concerns were raised by Fu et al., who analyzed real-world AI-generated code and found vulnerability rates exceeding 30% [3].

Given these concerns, this study aims to evaluate the trade-offs between productivity and security in AI-assisted development. The research will combine a developer survey [6], [7] with security analysis of AI-generated code [3], [8]. The findings will provide guidance for balancing efficiency and security in AI-driven coding workflows.

II. Research Questions

This study aims to answer the following research questions:

- **RQ1:** What security risks emerge in AI-generated code?
- **RQ2:** How can developers balance efficiency and security when using AI tools?
- **RQ3:** How do different AI-assisted coding tools compare in terms of security vulnerabilities?

A structured investigation will provide insights into optimizing AI-assisted development while mitigating risks.

III. Related Work

A. Impact of AI on Developer Productivity

AI-powered coding assistants have been widely adopted, with studies suggesting significant productivity benefits. Microsoft Research found that developers using GitHub Copilot completed tasks 55.8% faster than those coding manually [1]. IBM's study on AI pair programming in an enterprise setting observed perceived productivity gains but noted variability based on task complexity and developer experience [2]. However, Uplevel Data Labs reported conflicting results, stating that Copilot users introduced more defects while their overall task completion rates remained unchanged [5]. This discrepancy highlights the need for further investigation into the actual impact of AI coding assistants in real-world scenarios.

B. Security Risks in AI-Generated Code

Security concerns in AI-generated code have been extensively documented. One of the first empirical evaluations of GitHub Copilot revealed that approximately 40% of AI-generated code contained vulnerabilities, including SQL injection and buffer overflow issues [3]. Perry et al. found that AI-assisted developers wrote significantly more insecure code than those without AI assistance, despite being more confident in their solutions [4]. Additionally, Asare et al. demonstrated that AI-generated code often replicated past vulnerabilities, repeating the same mistakes found in historical software security incidents [8]. These findings suggest that while AI tools accelerate development, they also introduce new security risks that must be mitigated.

C. Developer Adoption and Perceptions

Despite security risks, the adoption of AI coding assistants has grown rapidly. A 2023 survey by GitHub reported that over 70% of developers believed Copilot helped them stay focused and avoid mental fatigue [6]. However, Snyk's industry report found that while 75% of developers believed AI-generated code was more secure than human-written code, 56% of respondents admitted to encountering security issues in AI-generated suggestions [7]. These findings indicate a potential **overconfidence bias** in AI-assisted development, where developers may unknowingly trust insecure AI-generated code.

D. Research Gaps

While multiple studies have examined AI's impact on productivity and security, there remains a lack of research that systematically evaluates both aspects together. Most existing studies either focus on productivity gains or security vulnerabilities in isolation. Additionally, prior work has not explored mitigation strategies for balancing efficiency and security when using AI-generated code. This study aims to bridge this gap by evaluating both **productivity and security concerns simultaneously** while identifying best practices for responsible AI-assisted development.

IV. Methodology

This study employs a mixed-methods approach to analyze the trade-offs between productivity and security in AI-assisted software development.

A. Research Method

A combination of surveys and code analysis will be used. Developer surveys have been widely used to assess AI tool adoption and perceived productivity benefits [6], [7]. Similarly, security analysis of AI-generated code has been employed in prior studies to assess vulnerability rates in AI-assisted programming [3], [4], [8].

B. Data Collection

- **Developer Survey:** An online survey with 40-50 software developers focusing on AI tool usage, productivity impacts, and security awareness. The survey structure is inspired by prior studies on developer adoption of AI tools [6], [7].
- **Code Analysis:** Examination of 25-30 AI-generated code samples for security vulnerabilities, following methodologies used in existing research on Copilot-generated code security risks [3], [8].

C. Data Analysis

Survey responses will be analyzed using statistical methods to identify patterns in productivity perceptions and security concerns. Code analysis will involve security testing using automated vulnerability detection tools, similar to approaches used in studies evaluating AI-generated security flaws [3], [4].

D. Threats to Validity

- **Internal Validity:** Potential biases in self-reported survey responses.
- **External Validity:** The extent to which findings generalize to diverse development environments.
- **Construct Validity:** Ensuring that productivity and security are measured using established frameworks from prior research [3], [6].

V. Acknowledgements

This study was a collaborative effort by **Muhammed Ali Elhasan, Yousef Abbas, and Zakaria Aslan**. Muhammed led the methodology and data analysis sections, Yousef contributed to the related work and research questions, and Zakaria handled the introduction and discussion of research gaps. We acknowledge the valuable feedback from our research advisors and colleagues who provided insights that helped refine our study.

VI. Change Log

This section highlights the changes made from Assignment 1 (A1) to Assignment 2 (A2).

- **Abstract Added:** A structured four-sentence abstract was included, as required in A2.
- **Expanded Methodology:** The methodology section was significantly improved by specifying the research method, participant selection, data collection techniques, data analysis approach, and threats to validity.
- **Updated Related Work:** Additional references were included to better contextualize the research and identify gaps in prior studies.
- **New Research Questions:** Update the research questions to better align with the study's objectives and scope.
- **Acknowledgements Section Added:** This new section specifies contributions of each author to the study.
- **Bibliography in IEEE Format:** References were reformatted to IEEE citation style and stored in a separate BibTeX file.

References

[1] S. Peng and M. Research, "The impact of ai on developer productivity: Evidence from github copilot," *arXiv*, 2023. [Online]. Available: https://arxiv.org/abs/2302.06590.

[2] J. Weisz and I. Research, "Examining the use and impact of an ai code assistant on developer productivity and experience in the enterprise," *CHI EA '25*, 2025. [Online]. Available: https://arxiv.org/abs/2401.03000.

[3] Y. Fu and et al., "Security weaknesses of copilot-generated code in github," in *ACM TOSEM (to appear)*, 2024. [Online]. Available: https://arxiv.org/abs/2302.08000.

[4] N. Perry and et al., "Do users write more insecure code with ai assistants?" In *ACM CCS 2023*, 2023. [Online]. Available: https://arxiv.org/abs/2302.07000.

[5] U. D. Labs, "Can generative ai improve developer productivity?" *Visual Studio Magazine*, 2024. [Online]. Available: https://visualstudiomagazine.com/articles/2024/09/ai-productivity.aspx.

[6] GitHub, "Survey reveals ai's impact on the developer experience," *GitHub Blog*, 2023. [Online]. Available: https://github.blog/news-insights/research/survey-reveals-ais-impact-on-the-developer-experience/.

[7] Snyk, "Ai-generated code leads to security issues for most businesses: Report," *CIO Dive*, 2023. [Online]. Available: https://www.ciodive.com/news/security-issues-ai-generated-code-snyk/705900/.

[8] O. Asare and et al., "Is github's copilot as bad as humans at introducing vulnerabilities in code?" In *ArXiv Preprint*, 2024. [Online]. Available: https://arxiv.org/abs/2401.01000.