Investigating Generative AI's Impact on Software Organizations' Security Practices: A Multi-Case Study Using Gioia Methodology

David Kinnberg Hein

Department of Computer Science, Aalborg University, Selma Lagerlöfs Vej 300, 9220 Aalborg, Denmark

davidkh@cs.aau.dk

Abstract. The integration of generative AI in software organizations presents promising opportunities for automating and enhancing security activities; however, its practical impact on the security practices of software organizations and associated risks remains underexplored. This study investigates how generative AI can enhance software security practices through a multi-case study involving five software organizations. By conducting semi-structured interviews with developers and managers, we identified four security practices where generative AI can support threat assessment, security testing, operational management, and education & guidance. At the same time, we uncover sociotechnical risks that may impede adoption, including concerns about inadequate data management of AI, inaccurate output, and a lack of trust. Guided by the OWASP SAMM framework and rooted in the Gioia methodological approach, our findings are synthesized into a theoretical model of generative AI adoption impacting security practices. This model highlights both the perceived benefits and the risks organizations encounter. Our work provides insights for software organizations looking to adopt generative AI into security-related workflows and contributes a foundational understanding of how such tools are perceived in practice. The paper emphasizes the importance of balancing innovation with caution as software organizations increasingly incorporate generative AI within secure development.

Keywords: Security Practices, Generative AI, Multi-case study, Gioia Methodology.

1 Introduction

In recent years, software organizations have seen a notable rise in security threats and vulnerabilities [11]. With the current rise of cyberattacks and data breaches, there is an increasing need for software organizations to prioritize security as part of the development process. However, manual security activities can be time-consuming, prone to errors, and expensive, making it difficult for software organizations to identify and mitigate security risks effectively [12]. In response to this challenge, generative AI has emerged as a promising tool to enhance software development tasks and security

activities [11]. Generative AI can be trained to analyze software code, vulnerability management data, and security logs to identify potential security risks and prioritize remediation efforts [8]. By automating security activities, software organizations can reduce the risk of human error, accelerate the detection and mitigation of security threats, and enhance the overall security posture of their software systems [16]. There have even been attempts to develop tailored generative AI for security purposes [1]. However, it remains largely unexplored how software practitioners themselves perceive the usefulness of using generative AI to enhance their security practices, in combination with the risks associated with using these tools.
Bridging this gap, we conducted a multi-case study with five software organizations to extend our knowledge on how generative AI can be used to improve security practices and the associated risks. Against this backdrop, we formulate our research question as follows:
How can the adoption of generative AI impact software organizations' security practices?
We conduct a multi-case study with five software organizations, using the Gioia methodology and the OWASP SAMM framework to categorize and link statements to security practices. Our study suggests that four security practices, Threat Assessment, Security Testing, Operational Management, and Education & Guidance, can be enhanced with generative AI, although sociotechnical risks pose challenges. We present a model integrating these factors to enhance security and address AI risks. The paper is organized as follows: Section 2 reviews related work on generative AI in security and risks; Section 3 details the research approach; Section 4 presents findings; and Sections 5 discuss these, limitations, and future directions.

2 Related Work

2.1 Generative AI in Software Development and Security

Previous research on generative AI in software development has explored its potential to assist developers with code generation, optimization, review, automated testing, and vulnerability scanning [25,27]. Automated code generation is well-established, with tools like GitHub Copilot evaluated for quality, though results vary. Effectiveness depends on factors such as programming language and task complexity [30]. These studies highlight the importance of human review of AI-generated code. Empirical evidence suggests that generative AI can enhance productivity by streamlining routine tasks [19]. Beyond code generation, AI assistance can improve developer knowledge, abilities, and productivity. Generative AI aims to enhance software engineering by automating routine tasks, freeing developers to focus on creative work [2]. For security activities, generative AI aids risk analysis, multilingual code auditing, and software testing. Its ability to process multiple programming languages helps detect hidden connections in architectures and source code, supporting accurate risk assessments

[16]. In testing, it creates test cases from natural language and enables automation for broader coverage. In binary analysis, interactive AI gathers contextual information to better understand complex code [6]. Many studies discuss generative AI as useful for vulnerability detection and remediation [26], but it's more effective at fixing vulnerabilities than detecting them [35]. Its accuracy depends on the model and task complexity, so caution is needed in security-critical areas. Generative AI has also been found to be suitable for logging [22], penetration testing [17], and code reviews [31]. However, most research is survey-based, lacking evaluation of impact on specific security practices or analysis of security benefits versus risks.

2.2 Risks of Using Generative AI

Trust is a central theme and risk in studies of generative AI adoption. Research has explored factors influencing developer trust [22], designs for trust-enhancing interfaces [24], and the role of online communities in shaping trust perceptions [5]. Findings show that perceived risks directly influence trust [33], and some studies suggest that users attribute similar trustworthiness to AI-generated and human-generated content [18]. Transparency has been repeatedly identified as a key factor in improving trust [9]. Thus, we must investigate the risks of using generative AI, as we want to manage and address them to ease adoption for software organizations. Several studies have identified critical risks associated with generative AI in software development. Generative AI tools may leak sensitive data [3] and spread misinformation [35]. In 2024, McGraw identified ten prominent risks: recursive pollution, data debt, improper use, black box opacity, prompt manipulation, poisoned data, reproducibility economics, unclear data ownership, lack of model trustworthiness, and insufficient encoding integrity [23].

More specifically, for software developers, include the risks of generating insecure code containing backdoors and vulnerabilities, risks of data manipulation, and unintended data transfer to external entities [10]. Another concern arises from generative AI models pre-trained on external datasets, which may introduce intellectual property violations when proprietary prompts are used to further train these models. A mixed-method study has examined the factors influencing the adoption of generative AI tools in software development. The findings show that, while these tools promise clear advantages, their adoption is far from guaranteed. The study highlights *compatibility* with the software organization as the primary enabler. Thus, generative AI must integrate seamlessly into an organization's existing development workflows and align with its values to gain traction. The study also points to risks that may hinder adoption, including inaccurate outputs and embedded biases in generated results [29].

3 Research Approach and Case Selection

We adopted a multi-case study approach, investigating five cases to enhance the generalizability of our findings and facilitate data triangulation [28], as opposed to the more in-depth approach of a single case study [14]. Our findings are detailed and general, enabling the development of a theoretical model. We focus on exploring how software companies can automate security activities using generative AI. The five cases were selected for their diverse contexts and shared interest in enhancing security activities with generative AI. This range helps identify cross-case patterns, allowing greater generalization of results [13]. All companies had prior knowledge of generative AI and security. The selected companies have varying levels of knowledge and experience in generative AI and security practices, providing insights into the roles of software professionals across organizational structures. We assessed participants' familiarity with these areas through interviews and explored their current experience in their role. This helped evaluate their ability to answer our questions. Experience with generative AI was limited to using tools, not developing large language models. Their prior experience is summarized in Table 1. To ensure confidentiality, all identifiers were anonymized, sensitive details were omitted or paraphrased, and informed consent was obtained.

3.1 Data Collection and Analysis

We undertook a comprehensive data collection via semi-structured interviews with 22 respondents from five software organizations. Conducted as focus groups of 2-4 participants, lasting 25 to 70 minutes, these groups included managers and developers. Selecting both roles was strategic, recognizing their unique perspectives on generative AI's impact on security practices. Their roles provide distinct insights that enhance our study. We developed an interview guide to ensure consistency and thoroughly explore our research question. This guide structured the interviews, ensuring relevant questions were consistently asked. Throughout the interviews, note-taking helped streamline data transcription, capturing participants' responses and observations for analysis [28,34].

Table 1: Description of participants and the five cases.

Participant	*Case*	*Sector*	*Org. Size*	*Role*	*Years of Exp.*	*Exp/ genAI*	*Exp/ Security*
P1	The Bankers	Financial Sector	>4000	Back-End Developer	1	Low	Low
P2	The Bankers	Financial Sector	>4000	Back-End Developer	22	Low	Low
P3	The Bankers	Financial Sector	>4000	Front-End Developer	4	Low	Low
P4	The Bankers	Financial Sector	>4000	Architect	8	Low	Low
P5	The Bankers	Financial Sector	>4000	Security Director	2	Medium	High
P6	The Bankers	Financial Sector	>4000	IT-Operations Manager	5	Low	Medium

P7	The Educators	Education	>60	Back-End Developer	2	Low	Low
P8	The Educators	Education	>60	DevOps	1	High	High
P9	The Educators	Education	>60	DevOps	2	High	High
P10	The Educators	Education	>60	DevOps	3	High	High
P11	The Educators	Education	>60	CEO	2	Medium	Medium
P12	The Data Analysts	Software Vendors	>60	DevOps	4	Medium	Medium
P13	The Data Analysts	Software Vendors	>60	Back-End Developer	2	Low	Low
P14	The Data Analysts	Software Vendors	>60	CTO	1	Low	Medium
P15	The Data Analysts	Software Vendors	>60	Team Lead	1	Low	Medium
P16	The Software Lawyers	Lawyer Firms	>40	CEO	34	Low	Medium
P17	The Software Lawyers	Lawyer Firms	>40	Team Lead	3	Low	Medium
P18	The Software Lawyers	Lawyer Firms	>40	CCO	3	Low	Low
P19	The Software Lawyers	Lawyer Firms	>40	Back-End Developer	2	Low	Low
P20	The Software Lawyers	Lawyer Firms	>40	Front-End Developer	1	Low	Low
P21	The Consultants	Software Vendors	>20	CEO	10	Low	Low
P22	The Consultants	Software Vendors	>20	COO	3	Low	Low

Following data transcription, our analysis focused on cross-case comparisons to identify patterns and themes [13]. We used a grounded theory approach based on the Gioia methodology to enhance rigor and structure [15]. The Gioia method offers a systematic framework, ensuring disciplined exploration of patterns and themes in qualitative data [21]. We adopt a theory-driven approach [4] to build a solid foundation for understanding security practices in software organizations. Our thematic analysis follows six steps: 1) Deeply familiarize with the dataset. 2) Outline initial coding schemes (1st order concepts) to categorize data. 3) Derive themes (2nd order themes) from codes, identifying patterns. 4) Review and refine themes for coherence. 5) Categorize and define themes to reflect nuances. 6) Produce a report summarizing findings and data structures [15]. NVivo was used to code our qualitative data. In the Gioia methodology, the interpreter combines 1st order concepts and 2nd order themes into aggregate dimensions, which serve as data structure titles. The categorization of security practices in the OWASP SAMM framework informs these themes. The framework has five business functions: Governance, Design, Implementation, Verification, and Operations, each with three practices. Governance includes Strategy & Metrics, Policy & Compliance, Education & Guidance. Design covers Threat Assessment, Security Requirements, and Secure Architecture. Implementation includes Secure Build, Secure Deployment, and Defect Management. Verification entails Architecture Assessment, Requirements-driven Testing, and Security Testing. Operations comprise Incident Management, Environment Management, and Operational Management.

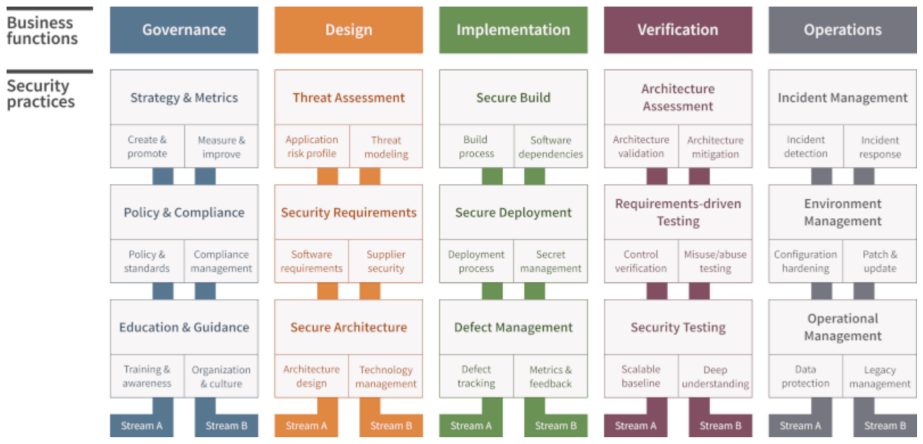

Fig 1: The OWASP SAMM framework showing the five business functions and their three security practices.

4 Findings

We have grouped our results into six data structures, comprising six distinct aggregate dimensions based on our 1st order concepts and 2nd order themes from the data analysis, as well as the security practices from OWASP SAMM.

4.1 Security Practices Improvement with Generative AI

Table 2 shows five security activities—Vulnerability scanning, repair, assessment, phishing attack simulation, and script analysis—that could benefit from generative AI across our five cases. Threat assessment involves identifying potential threats with tools to mitigate risks. Participants identified using AI for vulnerability scanning, with both developers and managers seeing it as crucial and feasible. A manager noted: "*In relation to security scanning of code, I believe that AI-technology can be used*" (P6). This suggests AI for code scanning is viable and viewed as a key security improvement. A developer, however, expressed doubts: "*It probably can, if you need to be kept up to* *date, then I think there are existing tools for scanning repositories in the background, because they are passive... the concept of constantly prompting does not seem relevant*" (P7). This implies that such tools could reduce cognitive load, but human validation remains necessary.

Table 2: Data Structure of the Aggregate Dimension: "Threat Assessment"

Quote	***1st Order Concepts***	***2nd Order Themes***	***Coverage (%)***
"Basically, getting notifications from Chat-GPT to tell us, well, you have such a system, and then you get a notification that on your system, you might be vulnerable to that. I can see that work." (P9)	Security code scan automation, increased effectiveness, reduced effort	Vulnerability scanning	55
"We became aware of a redux vulnerability. . .then asked ChatGPT to rewrite it without the vulnerability. That worked." (P3)	Fix vulnerability in code, security bug fixing, threat response, increased effectiveness, reduced effort,	Vulnerability repair	23
"I see value in having a language model synthesizing everything on your system and then getting you a clear message on, for example, you have this vulnerability on that machine, and then you should probably do that to resolve it." (P8)	Vulnerability classification optimization, increased effectiveness	Vulnerability assessment	18
"We can have them write custom spear phishing emails. . . and give it access to an API, so it can send an email from our domain." (P13)	Phishing attack automation, defence hardening, reduced effort, time saving	Phishing attack simulation	9
"Say we find a script on a machine, and we need to figure out how it works. They can deal with it quicker than a human analyst trying to comprehend the code. It would be an obvious way of using them." (P5)	Script analysis automation, reverse engineering, time saving	Script analysis	4,5

We report six security activities in operational management, including logging, data leakage prevention, access control, data flagging, flow documentation, and retrieval. Developers identified these activities, with logging being most prominent. A DevOps demonstrates using generative AI for automating log analysis: "*Let's say a log file in the cloud, saying, is there anything that is not of the ordinary in this, and it will actually* *be able to pass it much faster than we would be able to. So, I could see it being a good tool in the long run*" (P9). This shows a positive view of generative AI for this task, mainly improving security efficiency and freeing staff for proactive measures.

Table 3: Data Structure of the Aggregate Dimension: "Operational Management"

Quote	***1st Order Concepts***	***2nd Order Themes***	***Coverage (%)***
"I believe if you gave it access to log data, it could more quickly and effectively identify suspicious behavior." (P5)	Data logging automation, increased effectiveness	Logging	45
"I believe it is useful in many areas also for data leakage prevention." (P12)	Reduce data leakage, increased effective-ness, reduced effort	Data leakage prevention	23
"But I think it's obvious. Role configuration: which roles do we have, what do they have access to, and are there any holes?" (P14)	Improved role configuration, increased effectiveness, reduced effort	Access and permission control	14
"That's what they do all day; they get a ticket with all the data, and then you need to go through the whole ticket and confirm that nothing is wrong. So, yes, it could replace those centers specifically for that, you know, flagging." (P10)	Data analysis automation, task replacement, increased effectiveness	Data flagging	14
"I believe it can help us with documenting data flows." (P2)	Documentation automation, report and flow generation, increased effectiveness	Data flow documentation	9
"I could upload the message I received to ChatGPT and it could probably give me the data in 30 seconds instead of writing a SQL command." (P1)	Optimized data retrieval, convert text to SQL command, reduced effort	Data retrieval	4,5

Six security activities were identified: penetration testing, integration with static analysis tools, false-positive handling, secure code suggestions, unit testing, and code reviews. Security testing involves detecting vulnerabilities through techniques like static analysis, dynamic testing, penetration testing, and fuzz testing. It helps find weaknesses that attackers could exploit. Developers mainly identified these activities, with many noting penetration testing. A DevOps team member said, "*I could see those being used here as well as a helping hand in doing penetration testing*" (P12). Participants emphasized these outputs should supplement, not replace, human testing.

Table 4: Data Structure of the Aggregate Dimension: "Security Testing"

Quote	*1st Order Concepts*	*2nd Order Themes*	*Coverage (%)*
"I do know of tools that utilizes ChatGPT for guided penetration tests, where you have a tool that says, I want to attack this target, give me some suggestions on how to do it." (P10)	Attack scenario generation, test generation, improved penetration testing, increased effectiveness	Penetration testing	23
"It could make you more effective and faster, if it was integrated with static analysis tools." (P4)	Optimize static analysis tools, increased effectiveness, increased accuracy	Integration with static analysis tools	18
"So, not so skilled developers could be guided through a process of investigating false positives . . . Let's say he finds out that the false positive is present on the machine, then it can guide them through the process of figuring out if the issue is persistent, and how to remedy it." (P10)	Improved assessment of false positives, task replacement, increased effectiveness, in-creased accuracy, reduced effort	False positive handling	18
"Send it some code and ask if this is ok regarding security. I very much believe that's possible." (P1)	Improved security code assessment, code optimization, reduced effort	Secure code suggestions	14
"It dawned on me that they can help us with writing unit tests for our code." (P14)	Security test automation, test generation, in-creased effectiveness	Unit testing	9
"It can autogenerate code reviews." (P12)	Code review automation, reduced effort, increased effectiveness	Code reviews	4,5

Regarding the education and guidance practice, two activities were identified: acquiring knowledge of security threats and understanding security activities. These involve raising awareness and training employees, developers, and stakeholders. Participants highlighted generative AI as an educational resource, aiding staff in understanding threats and procedures, improving processes, and answering security questions. One manager noted its explanatory power: "*Some cyberattacks have funny names. It would be great at explaining them*" (P21). Others use it for reflecting on processes: "*Ask it for input on our security process and improvements. It should do that*" (P4). These accounts show that generative AI acts as a tool for automation and as an accessible knowledge partner for developers and managers.

Table 5: Data Structure of the Aggregate Dimension: "Education & Guidance"

Quote	*1st Order Concepts*	*2nd Order Themes*	*Coverage (%)*
"It would be a good idea to ask for advice, it would never hurt to ask for recommendations for improving our security model and approaches" (12)	Continuous knowledge generation, reduced effort, keeping up to date with cyber threats	Acquire knowledge of security threats	36
"You could ask it for input on our security process and ask for improvements. It should be able to do that." (P4)	Increased security knowledge acquisition, increased domain knowledge	Obtain understanding of security activities	27

4.2 Sociotechnical Risks of Using Generative AI

Despite optimism, participants expressed significant concerns about using generative AI for security, mainly 'sociotechnical' risks detailed in Table 6. The primary technical worry was 'Insufficient data management of AI,' cited by over half of the participants, involving uncertainty about how AI providers handle data. As an IT-operations manager said: "*How is the data stored? Is it trained to spit some of the same data out to somebody else?*"" This affects software professionals' willingness to use LLMs with sensitive data. A developer added that data could become dangerous if accessible to others. Lack of transparency into data storage raises privacy concerns and limits AI use in security tasks, given the sensitive data many organizations handle, like logs, social security numbers, and scripts. Participants also worried about AI providing inaccurate or fabricated outputs, with such issues noted in all cases. Other technical risks included data poisoning, outdated knowledge, and potential 'jailbreaking' into unsafe responses, revealing tension between AI's efficiency and its fragility as a security tool.

Table 6: Data Structure of the Aggregate Dimension: "Sociotechnical Risks"

Quote	*1st Order Concepts*	*2nd Order Themes*	*Coverage (%)*
"I would probably not use it to send code the other way around. So, there is an aspect of trust." (P2)	Hesitant towards generative AI, privacy concerns, accuracy concerns	Lack of trust	68
"Who is behind the AI is one thing, but it remains a black box how it prioritizes and how it manages your data." (P5)	Difficult to decipher how Generative AI work, privacy concerns	Lack of transparency	64
"The only big issue I see there is, again, how is the data stored? Is it trained to spit some	Unsatisfactory data storage and management,	Insufficient data management of AI	59

of the same data out to somebody else? That's the main concern when using large language models in general." (P6)	privacy concerns, sensitive data		
"Generative AI pose a security risk in itself because we have no control over what they send elsewhere." (P6)	Limited power over generative AI	Lack of control	45
"I think people are nervous about using it right now, because we don't have regulations for it yet." (P2)	Legal concerns, sensitive data, insufficient policies	Lack of regulations	36
"We have all seen many examples of it fabricating convincing answers that are wrong." (P4)	Inaccurate responses, requires assessment of output. lacking reliability	Inaccurate output of AI	32
"I am afraid of people, hackers or states systematically poisoning the well, and influencing the model's scales and responses." (P12)	Potential misuse application, potential for hackers	Data Poisoning	18
"I'm missing the last two years of data when I'm using ChatGPT and that's very irritating, especially if I start asking ChatGPT questions about Terraform." (P16)	Missing the newest data, limitation of use, lacking reliability	Lack of data novelty	14
"I am afraid that within 10 years it has taken over my job." (P3)	Worried about losing job, competitor for developer	Threat to job security	14
"It's still very new to me, what can they do and what can it not do?" (P3)	Limited knowledge of use, learning curve	Lack of competencies	9
"I remember that ChatGPT would not answer, but if it was asked to answer through a poem it would help you with suspicious things. If you can cheat ChatGPT to give you information about something it potentially becomes very dangerous." (P1)	Circumventing security protocols, possible to access prohibited information	Susceptible to jailbreaking	4,5
"If such a model turns on you instead of being a protective factor, exploiting the information you give and weaponizing it. That could be bad." (P22)	Potential of generative AI having ill intentions, reluctant to use for sensitive tasks	Malicious AI	4,5

More than half of the participants expressed concerns about 'lack of trust' and 'lack of transparency' affecting all five cases. Trust in the organization is essential for using service providers, especially as software engineers hesitate to rely on generative AI for security tasks due to mistrust. This distrust centers on privacy and the accuracy of AI, and hampers adoption. Participants also noted that models operate as a "*black box*" (P5), and additional concerns included a lack of regulations, job security fears, and skill gaps. These issues show that AI adoption involves organizational culture, governance, and perceptions, not just technology.

In summary, we highlighted the perceived benefits of using generative AI to improve security practices in software organizations. We focus on its potential to boost efficiency in areas like "Threat Assessment," "Operational Management," "Security Testing," and "Education & Guidance" through automation and increased productivity. However, our study also revealed concerns among software practitioners about "Sociotechnical Risks" linked to generative AI. Key technical risks include poor data management, inaccurate results, data poisoning, and limited data novelty. Additionally, social risks such as lack of trust, transparency, control, and regulation were identified. For practitioners, this mix of potential security improvements and risks underscores the need to carefully balance AI-driven efficiency with effective risk management and governance.

4.3 Theoretical Model and Hypotheses Formulation

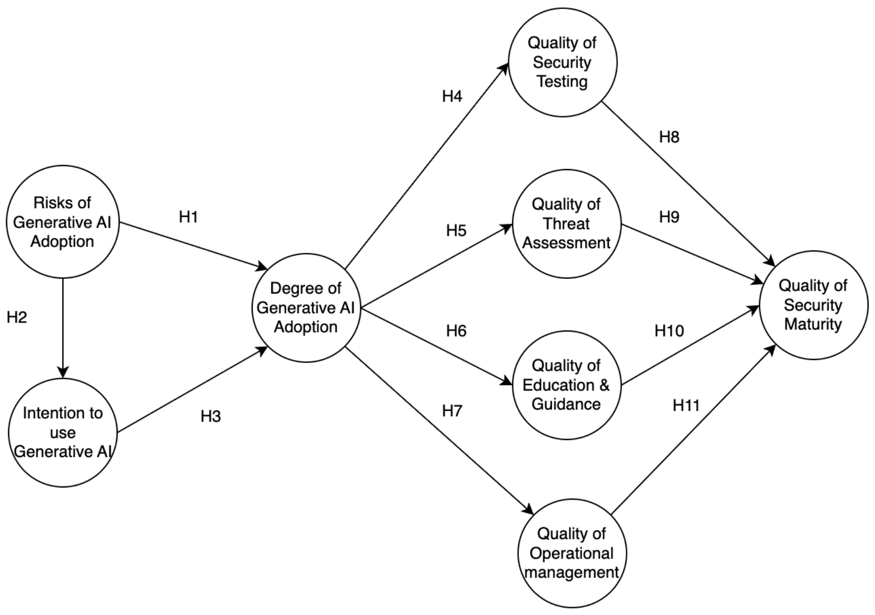
Quality of Security Testing
Quality of Threat Assessment
Quality of Education & Guidance
Quality of Operational management
Risks of Generative AI Adoption
Intention to use Generative AI
Degree of Generative AI Adoption
Quality of Security Maturity
H1
H2
H3
H4
H5
H6
H7
H8
H9
H10
H11

Fig 2: Theoretical model of Generative AI adoption for security practices

The findings laid the foundation for our conceptual model, depicted in Figure 1. The data structures from the findings section constitute our constructs. Four constructs serve as indicators of perceived improvements in security practices using generative AI. These constructs include Threat and Risk Assessment, Operational Management, Security Testing, and Education & Guidance. We added "Quality" in front of the four constructs to facilitate validation with quantitative methods. The final data structure, Sociotechnical Risks, was combined into the construct "Risks of using Generative". The construct "Intention to use generative AI" was drawn from prior work [7,29,32]. Hence, we focus on both the positive and negative impact of generative AI on security practices, formulated as hypotheses among the six constructs.

Drawing upon our qualitative results and general knowledge of software security and generative AI, we propose the following hypotheses:

Hypothesis H1: Risks of Generative AI adoption negatively impact software organizations' intention to use generative AI.

Technical risks of adopting generative AI in software organizations include insufficient data management, inaccurate output, malicious use of AI, lack of data novelty, susceptibility to jailbreaking, and malicious use of generative AI. These risks hinder the adoption of generative AI for security activities.

Social-related risks associated with adopting generative AI for security activities include uncertainties such as a lack of trust, transparency, control, and regulations, as well as threats to job security and competencies. Through qualitative theory generation, we observed that these risks can hinder software engineers from viewing generative AI as an appropriate tool for enhancing security practices. As a result, we formulate the following hypothesis:

Hypothesis. H2: Risks of generative AI adoption negatively impact the degree of generative AI adoption in software organizations.

Hypothesis H3: Intention to use positively impacts the degree of generative AI adoption in software organizations.

Hypothesis. H4: The degree of generative AI adoption has a positive impact on security testing.

The hypotheses have been formulated based on our analysis, which identified five different security activities for the security practice of security testing: logging, penetration testing, integration with static analysis tools, false positive assessment, and unit testing.

Hypothesis H5: The degree of generative AI adoption positively impacts the quality of threat assessment.

For the threat assessment security practice, we identified five distinct security activities: vulnerability scanning, vulnerability repair, vulnerability assessment, phishing attack simulation, and script analysis.

Hypothesis H6: The degree of generative AI adoption positively impacts the quality of education and guidance.

In relation to the education and guidance security practice, we identified two security activities: acquiring knowledge of security threats and obtaining knowledge of security activities.
Hypothesis H7: The degree of generative AI adoption positively impacts the quality of operational management.
Operational management identified five security activities that could be improved by adopting generative AI: data leakage prevention, access and permission control, data flagging, data flow documentation, and data retrieval. Building on these findings, we propose the following:
Hypothesis. H8: Utilizing generative AI for security testing has a positive impact on the quality of security maturity.
Hypothesis H9: Employing generative AI for threat assessment has a positive impact on the quality of security maturity.
Hypothesis H10: Adapting generative AI for education and guidance positively impacts the quality of security maturity.
Hypothesis H11: Adopting generative AI for operational management has a positive impact on the quality of security maturity.

5 Discussion

5.1 Contributions to Research

This study contributes to software engineering research at the intersection of generative AI and software security by addressing two critical and previously underexplored dimensions: the risks associated with generative AI tools and their potential for optimizing security practices. While previous work has examined generative AI in terms of productivity gains or code generation quality [2,20,29], our research extends this conversation to include its role in enhancing security practices and the obstacles to using these tools from a purely qualitative approach.
We build on existing research on generative AI risks [10,23,30] by identifying sociotechnical risks, such as a lack of trust, inadequate data management, inaccurate output, and a threat to developers. Using a qualitative multi-case study, we explore practitioners' perceptions, providing an empirical view of security risks. We also demonstrate how generative AI tools support threat assessment, operational management, security testing, and education, guided by the OWASP SAMM framework and Gioia methodology, showing where AI complements security processes. Additionally, we propose a theoretical model linking AI adoption risks and benefits, highlighting the roles of organizational enablers, perceived benefits, and risks. Based on detailed qualitative data analysis, the model offers hypotheses for future research.

5.2 Implications for Research

Our study offers practice implications, highlighting that while Generative AI shows promise in tasks like secure coding and threat modeling, organizations must use these tools carefully with a clear understanding of risks. Practitioners shouldn't assume AI output is secure; validation is necessary to ensure quality and security compliance. Organizations should assess their security maturity and governance before adopting AI tools. Based on the OWASP SAMM framework, successful integration depends on technical readiness and organizational alignment. Policies, training, and guidelines are key to managing risks like inaccurate output, accountability issues, and data exposure. By considering enablers, benefits, uncertainties, and risks, software teams can better focus efforts and engage stakeholders, supporting informed decisions and reducing resistance. Security leaders should monitor evolving regulations and ethics, since rapid AI development and weak governance require an adaptive risk approach for secure, responsible use.

5.3 Limitations and Future Research

Our study has limitations. A multi-case design offers cross-context insights but limits depth within each case. Focusing longer on fewer organizations could have yielded a richer understanding and observed changes over time. The study relies solely on qualitative data from semi-structured interviews, which captures perceptions but lacks generalizability and statistical rigor. Future research should use surveys or experiments to validate the model across broader samples. Although we included diverse industries and sizes, all companies are from Denmark, potentially biasing applicability to other regions. Lastly, participant perceptions reflect early generative AI adoption, and evolving tools and practices may reveal new risks and opportunities beyond this study. The key future research is empirically validating the theoretical model. While based on rich qualitative data, it needs quantitative testing with larger samples for generalizability and statistical robustness.

Future work involves structural equation modeling to test hypotheses and increase sample size. This validation could clarify causal links between enablers, risks, and adoption outcomes, advancing understanding of generative AI in security. Further research should examine model applicability across various organizational sizes, industries, and experience levels to better understand adoption dynamics and risks.

References

[1] E. Aghaei, X. Niu, W. Shadid, E. Al-Shaer, SecureBERT: A Domain-Specific Language Model for Cybersecurity, in: F. Li, K. Liang, Z. Lin, S.K. Katsikas (Eds.), Security and Privacy in Communication Networks, Springer Nature Switzerland, Cham, 2023: pp. 39–56.

[2] C. Bird, D. Ford, T. Zimmermann, N. Forsgren, E. Kalliamvakou, T. Lowdermilk, I. Gazit, Taking Flight with Copilot: Early insights and opportunities of AI-powered pair-programming tools, Queue 20 (2022) 35–57.

[3] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, A. Oprea, C. Raffel, Extracting Training Data from Large Language Models, (2021).

[4] P.J. Cash, Developing theory-driven design research, Design Studies 56 (2018) 84–119.

[5] R. Cheng, R. Wang, T. Zimmermann, D. Ford, "It would work for me too": How Online Communities Shape Software Developers' Trust in AI-Powered Code Generation Tools, (2023).

[6] S.S. Das, A. Dutta, S. Purohit, E. Serra, M. Halappanavar, A. Pothen, Towards Automatic Mapping of Vulnerabilities to Attack Patterns using Large Language Models, in: 2022 IEEE International Symposium on Technologies for Homeland Security (HST), 2022: pp. 1–7.

[7] F.D. Davis, Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology, MIS Quarterly 13 (1989) 319–340.

[8] A. Ding, G. Li, X. Yi, X. Lin, J. Li, C. Zhang, Generative Artificial Intelligence for Software Security Analysis: Fundamentals, Applications, and Challenges, IEEE Software (2024) 1–8.

[9] J. Drozdal, J. Weisz, D. Wang, G. Dass, B. Yao, C. Zhao, M. Muller, L. Ju, H. Su, Trust in AutoML: exploring information needs for establishing trust in automated machine learning systems, in: Proceedings of the 25th International Conference on Intelligent User Interfaces, ACM, Cagliari Italy, 2020: pp. 297–307.

[10] C. Ebert, J.P. Arockiasamy, L. Hettich, M. Weyrich, Hints for Generative AI Software Development, IEEE Software 41 (2024) 24–33.

[11] C. Ebert, M. Beck, Artificial Intelligence for Cybersecurity, IEEE Software 40 (2023) 27–34.

[12] C. Ebert, P. Louridas, Generative AI for Software Practitioners, IEEE Software 40 (2023) 30–38.

[13] K.M. Eisenhardt, T.E. Ott, Rigor in Theory Building from Multiple Cases, in: The Routledge Companion to Qualitative Research in Organization Studies, Routledge, 2017.

[14] B. Flyvbjerg, Five Misunderstandings About Case-Study Research, Qualitative Inquiry 12 (2006) 219–245.

[15] D.A. Gioia, K.G. Corley, A.L. Hamilton, Seeking Qualitative Rigor in Inductive Research: Notes on the Gioia Methodology, Organizational Research Methods 16 (2013) 15–31.

[16] M. Gupta, C. Akiri, K. Aryal, E. Parker, L. Praharaj, From ChatGPT to ThreatGPT: Impact of Generative AI in Cybersecurity and Privacy, IEEE Access 11 (2023) 80218–80245.

[17] A. Happe, J. Cito, Getting pwn'd by AI: Penetration Testing with Large Language Models, in: Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ACM, San Francisco CA USA, 2023: pp. 2082–2086.

[18] M. Huschens, M. Briesch, D. Sobania, F. Rothlauf, Do You Trust ChatGPT? -- Perceived Credibility of Human and AI-Generated Content, (2023).

[19] S. Imai, Is GitHub copilot a substitute for human pair-programming? an empirical study, in: Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings, Association for Computing Machinery, New York, NY, USA, 2022: pp. 319–321.

[20] Y. Liu, T. Le-Cong, R. Widyasari, C. Tantithamthavorn, L. Li, X.-B.D. Le, D. Lo, Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues, ACM Trans. Softw. Eng. Methodol. 33 (2024) 116:1-116:26.

[21] G. Magnani, D. Gioia, Using the Gioia Methodology in international business and entrepreneurship research, International Business Review 32 (2023) 102097.

[22] A. Mastropaolo, L. Pascarella, G. Bavota, Using deep learning to generate complete log statements, in: Proceedings of the 44th International Conference on Software Engineering, ACM, Pittsburgh Pennsylvania, 2022: pp. 2279–2290.

[23] G. McGraw, R. Bonett, H. Figueroa, K. McMahon, 23 Security Risks in Black-Box Large Language Model Foundation Models, Computer 57 (2024) 160–164.

[24] M. Mylrea, N. Robinson, AI Trust Framework and Maturity Model: Improving Security, Ethics and Trust in AI, Cybersecurity and Innovative Technology Journal 1 (2023) 1–15.

[25] K. Napier, T. Bhowmik, S. Wang, An empirical study of text-based machine learning models for vulnerability detection, Empir Software Eng 28 (2023) 38.

[26] H. Pearce, B. Tan, B. Ahmad, R. Karri, B. Dolan-Gavitt, Examining Zero-Shot Vulnerability Repair with Large Language Models, (2022).

[27] S.I. Ross, F. Martinez, S. Houde, M. Muller, J.D. Weisz, The Programmer's Assistant: Conversational Interaction with a Large Language Model for Software Development, in: Proceedings of the 28th International Conference on Intelligent User Interfaces, Association for Computing Machinery, New York, NY, USA, 2023: pp. 491–514.

[28] P. Runeson, M. Host, A. Rainer, B. Regnell, Case Study Research in Software Engineering: Guidelines and Examples, John Wiley & Sons, 2012.

[29] D. Russo, Navigating the Complexity of Generative AI Adoption in Software Engineering, ACM Trans. Softw. Eng. Methodol. 33 (2024) 135:1-135:50.

[30] A. Sergeyuk, Y. Golubev, T. Bryksin, I. Ahmed, Using AI-based coding assistants in practice: State of affairs, perceptions, and ways forward, Information and Software Technology 178 (2025) 107610.

[31] O.B. Sghaier, H. Sahraoui, A Multi-Step Learning Approach to Assist Code Review, in: 2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 2023: pp. 450–460.

[32] V. Venkatesh, J.Y.L. Thong, Xin Xu, Unified Theory of Acceptance and Use of Technology: A Synthesis and the Road Ahead, Journal of the Association for Information Systems 17 (2016) 328–376.

[33] S. Wang, L. Huang, A. Gao, J. Ge, T. Zhang, H. Feng, I. Satyarth, M. Li, H. Zhang, V. Ng, Machine/Deep Learning for Software Engineering: A Systematic Literature Review, IEEE Transactions on Software Engineering 49 (2023) 1188–1231.

[34] R.K. Yin, Case study research and applications: design and methods, Sixth edition, SAGE, Los Angeles, 2018.

[35] N. Zhang, L. Li, X. Chen, S. Deng, Z. Bi, C. Tan, F. Huang, H. Chen, Differentiable Prompt Makes Pre-trained Language Models Better Few-shot Learners, (2022).