Journal of Informatics and Web Engineering
Vol. 4 No. 2 (June 2025) eISSN: 2821-370X

Cyber Security Threats of Using Generative Artificial Intelligence in Source Code Management

Sainag Nethala[1]*, Sandeep Kampa[2], Srinivas Reddy Kosna[3]
[1,2,3] Cisco Systems, 3098 Olsen Dr, San Jose, CA 95128 USA
Corresponding author: (nethalasainag@gmail.com; ORCID: 0009-0002-1180-4058)

Abstract – Generative Artificial Intelligence (Generative AI) models are now broadly used for academic writing and software development for the sake of productivity and efficiency. Concerns on the impact of Artificial Intelligence (AI) tools on academic integrity and cybersecurity grow bigger with time. Generative AI is being used for code generation, editing, and review, raising ethical and security challenges. A big concern is the involuntary introduction of vulnerabilities into codebases. They can reproduce known bugs or malicious code that compromise software integrity because of the way models are trained: on large datasets. The tools may also pose additional security threats often encountered during software development. AI models trained on public data will generate code that resembles copyrighted content, creating ownership and legal grey areas. Use of AI to delegate coding increases potential adversarial attacks and model poisoning. Addressing these challenges would therefore call for a balanced approach towards AI integrating into software development. Secure coding practices, thorough testing, continuous monitoring, and collaboration between developers, security professionals, and AI researchers should be balanced. Strong governance, regular audits, transparency in AI development, and the embedding of ethical standards in AI usage will help in ensuring it is safe and effective. Generative AI should be seen as a tool to enhance, not replace, human expertise in software development. While automation can streamline workflows, developers must remain vigilant to detect and mitigate AI-induced vulnerabilities. A proactive approach that combines human oversight with AI-driven efficiency will be key to securing the future of software development.

Keywords—Generative AI, Source Code Management (SCM), Cyber Security Threats, AI-Generated Code Vulnerabilities, Code Injection Attacks, Data Poisoning

Received: 11 January 2025; Accepted: 10 March 2025; Published: 16 June 2025

1. INTRODUCTION

Generative Artificial Intelligence (Generative AI) shas increased swiftly, which resulted in the development of source code in cyber security. The improvements of Artificial Intelligence (AI) models have been marked with human level

complexity of code generation. However, alongside opportunity, the danger of these kinds development is more than concerning, revealing the deceit of a trojan horse, a weapon which can effectively be employed in criminal activities [1]. One big worry is the possibility of generative AI creating malware and other types of malicious software like backdoors or exploits. Hackers can simply train such AI models with existing harmful code that can generate newer versions of the harmful code capable of evading traditional defences in cyberspace. The resultant code is a severe challenge to cyber security analysts who depend upon established procedures for the detection of threats. Once again what this means for the cyber security experts is that the hackers can also use generative AI to create valid-looking code somehow planted into a large system or programme that relies on what is valuable information [2].

A growing threat is the use of generative AI to perpetrate social engineering attacks. Cyber criminals personalize emails or chat prompts that look and feel far more legitimate and trustworthy [3], [4]. Such conversation manipulation allows the fraudsters to ducer into divulging sensitive information or encouraging them to perform an action that gives up system security [1]. Such an ability to produce nearly convincing messaging poses significant challenges for individuals and organizations to identify a fraudulent transaction or communication [5], [6]. Therefore, both users and companies are now at greater risk of becoming victims of phishing attacks and data breaches [7]. The coupling of generative AI to manage source code does greatly raise issues of intellectual property rights and academic honesty [8] . Ambiguities remain when it comes to the ethical use of codes made by an AI without a solid mechanism in place [9]; this poses severe legal and moral predicaments for developers and organizations [10]. Figure 1 demonstrates diverse elements which help explain the crucial role of ai in software development. Many industry reports and research studies focus on describing the cybersecurity challenges that generative AI introduces when managing source code. Studies from the past have demonstrated these weaknesses as significant issues. Through practical case studies combined with expert insights the ideas are supported. Researcher demonstrates that AI-generated code poses security risks according to scientific literature [10]. Scientific research by Carlini et al. [11] showcased how viable adversarial attacks function against AI-produced code.

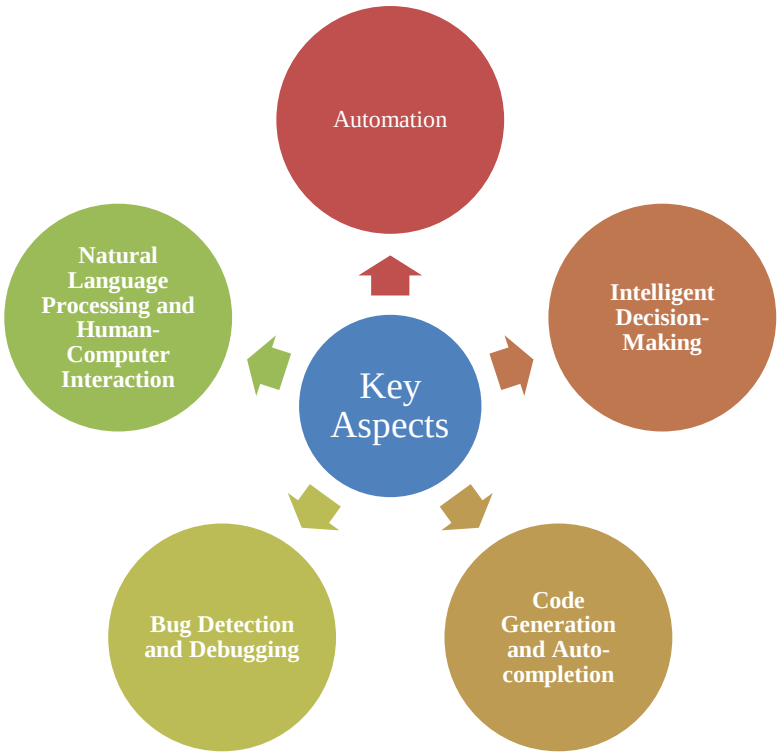

Figure 1. Different Factors to Understand the Significance of AI in Software Development.

2. LITERATURE REVIEW

2.1 Background

A multi-stage research method helped identify cybersecurity threats and report them in connection to generative AI within source code administration systems. The methodology unites literature review with empirical threat validation along with expert analysis in order to identify threats in their entirety.

2.1.1 Literature Review-Based Threat Identification

Various cybersecurity studies about code vulnerabilities generated by AI systems underwent systematic evaluation through literature review methods. A review of research papers in IEEE ACM Springer Elsevier and security white papers allowed for the identification of documented threats. Review of peer-reviewed research in this phase led to the identification of threats that passed validation tests. The essential works related to AI-generated code vulnerabilities in source code are represented by [1], [11], [12].

2.1.2 Empirical Threat Analysis through AI Model Testing

Experimental testing of AI-powered code-generation models including OpenAI's Codex and GitHub Copilot and Meta's CodeLlama took place in order to validate the identified threats. Security analysts used their expertise to find problems in coding practices that were unsafe and discover backdoor entries and data leaks in systems. AI-generated code outputs underwent analysis to detect three security risks: (1) vulnerable included dependencies; (2) input manipulation flaws; and (3) unanticipated security flaws. The testing approach used here matches techniques defined in which reveal actual AI system weaknesses.

2.1.3 Expert Validation of AI-Generated Threats

Specialists from the field of AI-driven development and DevSecOps confirmed the security threats through expert opinions. Information from structured expert interviews and security reports produced by Open Worldwide Application Security Project (OWASP), National Institute of Standards and Technology (NIST) and MITRE ATT&CK framework was used for this study. Vital insights helped validate that the recognised threats were not basic theoretical concepts yet shown they existed in real-world industrial applications [13].

2.1.4 Threat Categorization and Risk Assessment

The evaluators validated threats that had undergone identification which then were positioned into classification categories based on their severity level and probability of exploitation. The developed threat classification system used elements from MITRE ATT&CK and NIST SP 800-53 as well as OWASP AI Security Guidelines. The project evaluated high-risk threats starting with model poisoning followed by both backdoor attacks and data leakage according to their risks for source code security breakdown [14], [15].

2.2 Related Work

Generative AI in source code management creates cyber security challenges as traditional methods cannot address its vulnerabilities cc. These models write code according to instruction but can expose sensitive information or create malicious code. The integration of AI increases its risk for malware, vulnerable code, or backdoors. Threat actors may manipulate outputs or poison training data causing unexpected outcomes [12]. A case for secure practices while using AI tools in software development is made clear with these risks [14].

Data privacy concerns arise with generative AI handling sensitive information like intellectual property, confidential code, and personal data, which are exposed to unauthorised users [16]. AI models trained on mixed datasets may leak private information unless there is stringent enforcement of security practices. Such a leakage may happen both during

training and deployment thereby becoming practically undetectable. There exists a vulnerability of collaborative software platforms toward social engineering attacks. Generative AI is now able to craft fake code that imitates access points or leaks credentials for unauthorised access. These threats serve as a reason to adopt stringent access controls and maintain secure data practices [11].

The intellectual property threat caused by generative AI is salient in the realm of attacks. The AI's code can produce products providing a degree of randomness, so that a training programme that is pre-organised can propagate reproducing and reflecting a company's proprietary algorithm again without providing citations or approved-from-rights notices [11]. As AI becomes more sophisticated through generating code resembling human-like codes such chances of incidental duplicity heighten [17]. Developers might repent on usages of codes under the pretext of AI oblivious of legitimacy, hence leading up to legal industrialism and avaricious losses [18]. Clear provisions need to be established so as to define proprietary rights and secure attribution.

Fans of generative AI continue on into supporting crime-in-process since it is needed to simplify and create malicious code or instantly manufacture attack scripts [19]. Huge-scale attacks may now be processed with little efforts thanks to attackers. Information generation, automation of phishing attacks, and security defences all threaten a new era of attacks. AI-generated phishing messages look much like other legitimate messages-and hence are difficult to distinguish from fraud [20]. This is turning up the possibility of attacks on computers and systems via social engineering causing data breaches and system compromises.

Such organizations must have strong cyber security strategies in place to deal with all of these risks. Such might involve the adoption of advanced threat detection tools and the running of regular audits, along with strict enforcement of data privacy policies [21]. Early detection of potential threats can come from continuous monitoring of AI outputs and validation of datasets. Developers should ensure secure coding practices and perform a rigorous code review to discover vulnerabilities that might be introduced by generative AI. Continuous monitoring and evolving, adaptable security frameworks will be needed as the AI technologies progress.

Investments in AI-fuelled cyber security education to inform developers about the risks involved with generative AI writing code for software development and establishing clear guidelines on ethical AI use [22]. Sound security controls and proactive practices enable organizations to benefit from generative AI while containing risks. A balanced approach, weighing both innovation and security, promotes the safe and responsible adoption of generative AI source code management. Table 1 provided key findings and security implications of AI in source code management.

Table 1. Key Findings and Security Implications of AI in Source Code Management.

Ref	Focus	Key Findings	Implications for Security
2023 [10]	Dependency risks in AI-generated code	Unverified dependencies pose critical vulnerabilities.	Outdated or insecure libraries can lead to system compromises.
2022 [11]	AI-powered automated code reviews	AI struggles to detect complex vulnerabilities.	Human oversight remains essential for effective code review.
2024 [12]	Adversarial attacks on AI models	AI models are vulnerable to adversarial manipulation.	Requires adversarial testing to prevent tampered code suggestions.
2022 [13]	Malicious code injection via AI	AI can suggest malicious code snippets.	Potential for unauthorised backdoors or malware in source code.
2023 [16]	AI-generated code vulnerabilities	AI-generated code often lacks robust error handling.	Increased risk of security flaws due to poor threat modelling.
2023 [21]	SCM workflow automation threats	AI-driven automation can bypass security checks.	Unvetted code deployments may lead to system breaches.
2021 [23]	Model poisoning in code generation	Model poisoning can inject malicious code.	Robust validation needed to prevent manipulated AI outputs.

3. RESEARCH METHODOLOGY

3.1 Threat and Vulnerabilities in Generative AI

3.1.1 Threat Prospective of Generative AI in Source Code

Literature reviews confirm that code generated by AI systems introduces vulnerabilities which affect entire systems. Agrafiotis et al. [12] together with Dell'Acqua et al. [23] demonstrate that AI software development lacks standard security practices. Research by Carlini et al. [11] demonstrates actual poisoning attack methods which target AI models because of training data manipulation problems. Pingree et al. together with expert work show that AI-driven development cannot proceed successfully without superior security measures. The use of Generative AI with sources from coding systems makes a hole for security threats. Attackers can utilize these models to generate malicious code like malware and exploits which risk breaking the code integrity. The Generative AI can create malicious Social Engineering attacks that may mislead users into clicking on a link or downloading a file [23]. This kind of AI presents easy opportunities for phishing attempts to initiate unauthorised access and injure data. The use of generative AI maintains concerns about intellectual property rights, academic integrity, and accountability obligations for the security of generated code. Without proper safety measurements, developers can integrate or unauthorised code into their projects [24]. Addressing these challenges requires strict supervision and cooperation strategies to ensure that the generated AI increases productivity without affecting security [25]. Table 2 shows the security threats and consequences of Generative AI in source code management.

Table 2. Security Threats and Consequences of Generative AI in Source Code Management.

Ref	Threats	Consequences
[25]	Security vulnerabilities	AI-generated code may introduce flaws, enabling malware, backdoors, or exploits.
[26]	Data privacy and confidentiality	Generative AI may expose trade secrets, proprietary code, or personal data.
[27]	Adversarial attacks	Training data manipulation can lead to malicious code generation.
[28]	Intellectual property theft	Generated code may replicate proprietary content without permission.
[29]	Malicious use of AI	AI can automate harmful code creation for cyberattacks and phishing.
[30]	Unintended consequences	AI may produce faulty outputs, requiring rigorous testing and monitoring.

The injection of generative AI into systems of source code management gives the main rise of cyber security risks. The adversary can succeed in applying these models for generating codes of malware and exploits attacking code integrity [26]. Alternatively, they may use generative AI to construct appealing social engineering attacks thus making it more challenging to distinguish between real and malicious communications [27]. Which opens widens the door to phishing, unauthorised access, and data breaches. When generative AI is applied This comes with intellectual property rights, as well as academic integrity and being accountable for the security of code that is generated. Without any form of security, developers may integrate vulnerable or even unauthorised code into their projects, thus addressing the challenges which should involve strict supervision and collaborative approaches with the end goal of making generative AI improve productivity without hampering security [28].

3.1.2 Vulnerabilities and Attack Surfaces of AI in Source Code

Generative AI models function like any other programme and can have vulnerabilities that are attacked by adversaries. These weaknesses leave systems open to severe threats if not handled well [29]. Hackers can manipulate AI to generate harmful code or access sensitive data [30], [31]. They often target the model's training process or input data to bypass security. Inadequate safeguards make AI-powered systems easier to target. The shared weaknesses of AI-based source code management are the following.

3.2. Input Manipulation

AI models are relayed on input data to give correct outputs. Attackers can manipulate these inputs to trick the system into creating wrong or malicious outputs. This achieved through carefully crafting input data to take advantage of the model's vulnerabilities [32]. Malware can also be hidden in the input to violate system integrity. Input validation and continuous monitoring are required to avoid these attacks [33].

3.3. Model Poisoning

AI models learn from patterns in large datasets. Attackers can inject malicious data to alter model behaviour and causing biased outputs or weakened defences. Poisoned models may overlook malicious code or dangerous patterns. These attacks are hard to detect as corrupted data blends with valid inputs. Strong dataset validation and secure training environments can reduce the risk of model poisoning [34].

3.4. Model Inversion

AI models learn from patterns which are present in datasets. Attackers can induce malicious data to modify model behaviour and cause biased outputs and weakened defences. Poisoned models may overlook malicious code or dangerous patterns. These attacks are hard to detect as corrupted data blends with valid inputs. Well-built dataset validation and secure training environments can reduce the risk of model poisoning [34].

3.5. Backdoor Attacks

The case for backdoor attacks lies in making inherent ill-functioning functions in AI models that remain suspect until the certain inputs trigger them. This will allow the intruder to bypass secure controls, giving one unauthorised access. This may lead to information theft, system manipulation, and complete system compromise. It is difficult to detect backdoors since they work normally under completely legitimate inputs until they are triggered. But since they are difficult to spot out, abuse cases must be reverted without a second delay in focusing completely on regular code auditing for backdoor identification and implementation [11].

3.6. Data Leakage

AI systems encounter security threats upon deployment even if the model is deemed secure. Weak access controls and open communication channels expose the systems to attacks. Poor data protection only makes the situation worse. In this case, hackers could utilize such vulnerabilities to access intrusive Uniform Resource Locators (URLs) or other altered outputs. Strong protocols and access controls are essential to protect AI systems from exploitation [35].

3.7. Insecure Deployment

Data leakage occurs when AI systems expose sensitive information due to weak data protection. This occurs when large data sets are stored without encryption. These gaps could be utilised by hackers to access private data, thus jeopardizing user privacy and leaving the organizations with legal and financial issues. The risk increases when AI deals with personal or proprietary information. Strong encryption, strict access controls and continuous monitoring are essential to prevent data leakage [36]. To address the vulnerabilities, developers should adopt secure coding practices and run adequate testing and validation routines. Attack prevention in the design and implementation of AI systems is possible through the integration of security controls. Among these are the proper cleaning of training data and input validation. Regular security audits can peel away hidden gaps before they have a chance of getting utilised. Continuous monitoring aids in identifying and responding to atypical behaviour with speed. These practices complicate the problem of maintaining the integrity and security of computer-generated source codes [37]. Table 3 provided key vulnerabilities and attack surfaces in ai-powered source code management with mitigation strategies.

Table 3. Key Vulnerabilities and Attack Surfaces in AI-Powered Source Code Management with Mitigation Strategies

Threat	Description	Mitigation Measures	Ref
Input Manipulation	Attackers craft inputs to exploit AI vulnerabilities, causing incorrect or malicious outputs. Malware can be embedded in input data.	Input validation, continuous monitoring, and strict filtering of inputs.	[38]
Model Poisoning	Injecting malicious data into training datasets to alter model behaviour, causing biased outputs or weakened security.	Strong dataset validation, secure training environments, and adversarial training.	[39]
Model Inversion	Exploiting model outputs to reconstruct training data, potentially exposing sensitive information.	Differential privacy, restricting model access, and limiting excessive querying.	[40]
Backdoor Attacks	Maliciously embedding hidden triggers in AI models, allowing unauthorised access when specific inputs are used.	Regular code audits, strict model validation, and adversarial testing.	[41]

4. DISCUSSIONS

The application of integrating generative AI into source code management systems entails one key factor in the incorporation of existing systems. This would provide the organization with automated code generation and suggestion services based on vast datasets collected from open repositories or developer's inputs. While these services increase productivity, they also present a threat to the cybersecurity of the organization's code integrity, system security, and IP [38]. One of the key risks involved is that AI generated code can be prone to considerable exploitation. Due to a generative AI model having been trained on existing data, it is more likely to reproduce flaws that already exist within the system. If AI tools suggest insecure code snippets, these tools are capable of propagating system vulnerabilities by reusing unsafe coding patterns. One such flaw, The Heartbleed bug, stems from human coding errors which made systems vulnerable, and AI powered tools could possibly replicate such negative practices without positive control [39]. Tools that utilize Generative AI models are also prone to adversarial attacks as there are no positive controls: attackers can easily interfere with training data or engage in activities with AI tools to obtain the desired result. This may result in producing malicious code like backdoors, malware or ransomware presented in the form of legitimate source code. In this scenario AI-driven pull requests can become approved by code reviews and bring along unknown dangers. Malware can be injected by the attackers into open-source projects widely used by organizations by modifying the functioning of AI models. This strategy takes advantage of the reliance that developers have on automated systems making it more challenging to identify security violations [40].

Intellectual property and licensing are another concern. AI models trained on public or private databases can produce code that mirrors copyrighted work, which could cause legal issues. Further, this could be the most problematic in open-source communities where adherence to licenses is vital. The contention around GitHub Copilot serves to suggest this risk as users reported cases where AI-written code has taken entire chunks from currently available repositories without any attribution being given. This means that AI-based SCM tools could unknowingly violate IP rights [41]. Improper usage of AI in code reviews also heightens the chances of breach in cyber security. While AI could flag up errant code with a few suggestions for revisions, it could also be used by attackers to obfuscate human interference. They can use AI-assisted code to generate vulnerabilities that could remain undetected or evade security testing. This risk is amplified in environments with a high reliance on automated systems to conduct code reviews, lowering the possibility of detection of malicious code before being deployed [15]. Such challenges can be overcome through comprehensive security measures employing AI capabilities with human expertise. Developers must carry out automated as well as manual reviews of AI-generated code to unearth any potential threats. Proper safeguards ensure that at the very least AI training data does not contain vulnerable or copyrighted code. Transparent development of AI tools, periodic audits of code-generation models, and transparency in AI-based contributions could further lower cybersecurity threats. Such steps allow the utilization of generative AI by organizations while also defending their source code management systems from forthcoming attacks [41].

5. CONCLUSION

Generative AI has ramped up source code management to a greater level of productivity, automation in coding, and code quality, while potentially challenging the organization's cyber security environment. The list of threats includes intellectual property theft, malicious code insertion, and unintended vulnerabilities threatening system integrity. Failure to mitigate these items has straight-and-to-the-point repercussions for an organization, offering an open, albeit dangerous new gate for data breaches, lawsuits, and system collapses all giving even greater urgency to put security first while welcoming AI-infused development tools. Solutions to the challenges facing the future will call for secure coding practices, extensive testing, and teamwork between developers, AI researchers, and security professionals. Use of clean training data, frequent updates to the models, and good access controls is also a safeguard to lower risk to some degree. While innovation is enjoyed, organizations can then adopt AI-powered source code management and secure their software systems when protecting against new cyber threats in a well-structured and secure manner.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their valuable comments.

FUNDING STATEMENT

The authors received no funding from any party for the research and publication of this article.

AUTHOR CONTRIBUTIONS

Sainag Nethala: Project Administration, Methodology, Data Curation, Validation, Writing – Review & Editing.
Sandeep Kampa: Conceptualization, Data Curation, Methodology, Validation, Writing – Original Draft Preparation.
Srinivas Reddy Kosna: Project Administration, Supervision, Writing – Review & Editing.

CONFLICT OF INTERESTS

The authors declare no conflict of interests.

ETHICS STATEMENTS

Our publication ethics follow The Committee of Publication Ethics (COPE) guideline. https://publicationethics.org/. No human subjects involved, no animal experiments involved, the work involved not data collected from social media platforms.

REFERENCES

[1] E. Ferrara, "GenAI against humanity: nefarious applications of generative artificial intelligence and large language models," *Journal of Computational Social Science*, vol. 7, no. 1, pp. 549–569, Apr. 2024, doi: 10.1007/s42001-024-00250-1.

[2] D. M. Huber and O. Alexy, "The impact of artificial intelligence on strategic leadership," in *Handbook of Research on Strategic Leadership in the Fourth Industrial Revolution*. Edward Elgar Publishing, 2024, pp. 108–136. [Online]. Available: https://www.elgaronline.com/edcollchap/book/9781802208818/book-part-9781802208818-12.xml.

[3] N. Inie, J. Falk, and S. Tanimoto, "Designing participatory AI: Creative professionals' worries and expectations about Generative AI," in *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. Hamburg, Germany: ACM, Apr. 2023, pp. 1–8, doi: 10.1145/3544549.3585657.

[4] G. W. Choi, S. H. Kim, D. Lee, and J. Moon, "Utilizing Generative AI for instructional design: Exploring strengths, weaknesses, opportunities, and threats," *TechTrends*, vol. 68, no. 4, pp. 832–844, Jul. 2024, doi: 10.1007/s11528-024-00967-w.

[5] R. Jain and A. Jain, "Generative AI in writing research papers: A new type of algorithmic bias and uncertainty in scholarly work," in *Intelligent Systems and Applications*, vol. 1065, K. Arai, Ed. Cham, Switzerland: Springer, 2024, pp. 656–669, doi: 10.1007/978-3-031-66329-1_42.

[6] H. Hessari, A. Bai, and F. Daneshmandi, "Generative AI: Boosting adaptability and reducing workplace overload," *Journal of Computer Information Systems*, pp. 1–14, Oct. 2024, doi: 10.1080/08874417.2024.2417672.

[7] E. Brynjolfsson, D. Li, and L. R. Raymond, "Generative AI at work," National Bureau of Economic Research, Working Paper 31161, 2023.

[8] K. Greshake et al., "Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection," in *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*. Copenhagen, Denmark: ACM, Nov. 2023, pp. 79–90, doi: 10.1145/3605764.3623985.

[9] D. Humphreys et al., "AI hype as a cyber security risk: The moral responsibility of implementing generative AI in business," *AI Ethics*, vol. 4, no. 3, pp. 791–804, Aug. 2024, doi: 10.1007/s43681-024-00443-4.

[10] S. Wen, "The power of generative AI in cybersecurity: Opportunities and challenges," *Applied Computer Engineering*, vol. 48, pp. 31–39, 2024.

[11] N. Carlini et al., "Poisoning web-scale training datasets is practical," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2024, pp. 407–425. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10646610.

[12] I. Agrafiotis et al., "A taxonomy of cyber-harms: Defining the impacts of cyber-attacks and understanding how they propagate," *Journal of Cybersecurity*, vol. 4, no. 1, p. tyy006, 2018.

[13] E. M. Renieris, D. Kiron, and S. Mills, "Building robust RAI programs as third-party AI tools proliferate," *MIT Sloan Management Review*, 2023. [Online]. Available: https://sloanreview.mit.edu/projects/building-robust-rai-programs-as-third-party-ai-tools-proliferate/.

[14] M. Taddeo, "Information warfare: A philosophical perspective," in *The Ethics of Information Technologies*. Routledge, 2020, pp. 461–476. [Online]. Available: https://www.taylorfrancis.com/chapters/edit/10.4324/9781003075011-35/information-warfare-philosophical-perspective-mariarosaria-taddeo.

[15] D. Schatz, R. Bashroush, and J. Wall, "Towards a more representative definition of cyber security," *Journal of Digital Forensics, Security and Law*, vol. 12, no. 2, p. 8, 2017.

[16] D.-O. Jaquet-Chiffelle and M. Loi, "Ethical and unethical hacking," in *Ethics in Cybersecurity*. Springer, 2020, pp. 179–204.

[17] S. Bukhari, B. Tan, and L. De Carli, "Distinguishing AI- and human-generated code: A case study," in *Proceedings of the 2023 Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*. Copenhagen, Denmark: ACM, Nov. 2023, pp. 17–25, doi: 10.1145/3605770.3625215.

[18] K. Lee, J. Lee, and K. Yim, "Classification and analysis of malicious code detection techniques based on the APT attack," *Applied Sciences*, vol. 13, no. 5, p. 2894, 2023.

[19] A. Odeh, N. Odeh, and A. S. Mohammed, "A comparative review of AI techniques for automated code generation in software development: Advancements, challenges, and future directions," *TEM Journal*, vol. 13, no. 1, p. 726, 2024.

[20] S. Ness et al., "Anomaly detection in network traffic using advanced machine learning techniques," *IEEE Access*, vol. 13, pp. 16133–16149, 2025, doi: 10.1109/ACCESS.2025.3526988.

[21] M. Sallam, "ChatGPT utility in healthcare education, research, and practice: Systematic review on the promising perspectives and valid concerns," *Healthcare*, vol. 11, no. 6, p. 887, 2023. [Online]. Available: https://www.mdpi.com/2227-9032/11/6/887.

[22] V. Shinde et al., "Ensemble voting for enhanced robustness in DarkNet traffic detection," *IEEE Access*, vol. 12, pp. 177064–177079, 2024, doi: 10.1109/ACCESS.2024.3489020.

[23] F. Dell'Acqua et al., "Navigating the jagged technological frontier: Field experimental evidence of the effects of AI on knowledge worker productivity and quality," Harvard Business School Technology & Operations Mgt. Unit Working Paper No. 24-013, 2023. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4573321.

[24] T. Shahzad and K. Aman, "Unveiling the efficacy of AI-based algorithms in phishing attack detection," *Journal of Information and Web Engineering*, vol. 3, no. 2, pp. 116–133, 2024.

[25] M. M. Rahman and Y. Watanobe, "ChatGPT for education and research: Opportunities, threats, and strategies," *Applied Sciences*, vol. 13, no. 9, p. 5783, 2023.

[26] OpenAI et al., "GPT-4o system card," arXiv:2410.21276, Oct. 2024. [Online]. Available: https://arxiv.org/abs/2410.21276.

[27] M. Gupta et al., "From ChatGPT to ThreatGPT: Impact of Generative AI in cybersecurity and privacy," *IEEE Access*, vol. 11, pp. 80218–80245, 2023.

[28] D. Su et al., "GPT store mining and analysis," arXiv:2405.10210, May 2024. [Online]. Available: https://arxiv.org/abs/2405.10210.

[29] K. M. Caramancion, "An exploration of disinformation as a cybersecurity threat," in *2020 3rd International Conference on Information and Computer Technologies (ICICT)*. IEEE, 2020, pp. 440–444, doi: 10.1109/ICICT50521.2020.00076.

[30] B. Edwards, "AI-powered Bing Chat spills its secrets via prompt injection attack," *Ars Technica*, 2023.

[31] T. Munusamy and T. Khodadi, "Building cyber resilience: Key factors for enhancing organizational cyber security," *Journal of Information and Web Engineering*, vol. 2, no. 2, pp. 59–71, 2023.

[32] L. De Angelis et al., "ChatGPT and the rise of large language models: The new AI-driven infodemic threat in public health," *Frontiers in Public Health*, vol. 11, p. 1166120, 2023.

[33] D. Bruschi and N. Diomede, "A framework for assessing AI ethics with applications to cybersecurity," *AI Ethics*, vol. 3, no. 1, pp. 65–72, Feb. 2023, doi: 10.1007/s43681-022-00162-8.

[34] M. Cascella et al., "Evaluating the feasibility of ChatGPT in healthcare: An analysis of multiple clinical and research scenarios," *Journal of Medical Systems*, vol. 47, no. 1, p. 33, Mar. 2023, doi: 10.1007/s10916-023-01925-4.

[35] R. Klemke and H. Jarodzka, "Locked in Generative AI: The impact of Large Language Models on educational freedom and teacher education," in *Exploring New Horizons in Generative Artificial Intelligence in Teacher Education*. Springer, 2024, p. 76.

[36] A. Varma, C. Dawkins, and K. Chaudhuri, "Artificial intelligence and people management: A critical assessment through the ethical lens," *Human Resource Management Review*, vol. 33, no. 1, p. 100923, 2023.

[37] S. F. Wamba et al., "Are both generative AI and ChatGPT game changers for 21st-century operations and supply chain excellence?," *International Journal of Production Economics*, vol. 265, p. 109015, 2023.

[38] B. C. Stahl and D. Eke, "The ethics of ChatGPT—Exploring the ethical issues of an emerging technology," *International Journal of Information Management*, vol. 74, p. 102700, 2024.

[39] R. A. Spinello, "Corporate data breaches: A moral and legal analysis," *Journal of Information Ethics*, vol. 30, no. 1, pp. 12–32, 2021.

[40] L. P. Robert et al., "Designing fair AI for managing employees in organizations: A review, critique, and design agenda," *Human–Computer Interaction*, vol. 35, no. 5–6, pp. 545–575, Nov. 2020, doi: 10.1080/07370024.2020.1735391.

[41] P. N. Petratos and A. Faccia, "Fake news, misinformation, disinformation and supply chain risks and disruptions: Risk management and resilience using blockchain," *Annals of Operations Research*, vol. 327, no. 2, pp. 735–762, Aug. 2023, doi: 10.1007/s10479-023-05242-4.

BIOGRAPHIES OF AUTHORS

	**Sainag Nethala** is a Splunk Assigned Expert, United States, functioning as a Principal Technical Specialist in Security & Observability Solutions. With over 9 years of experience pioneering machine learning applications in cybersecurity and enterprise observability, I combine deep technical expertise with strategic implementation. I architect and deliver advanced security and observability solutions while contributing to the broader technical community through research publications and thought leadership. He can be contacted at email: nethalasainag@gmail.com.
	**Sandeep Kampa** is a Senior DevSecOps Engineer at Splunk Inc., United States He holds a Master's in Computer Software Engineering from Stratford University and a Bachelor's in Technology from JNTU Hyderabad, India. Sandeep has led cross-functional teams, enhancing efficiency and reliability across various sectors. He has authored scholarly articles and served as a judge for hackathons and national competitions. Additionally, he volunteers as a Software Project Manager, mentoring communities in technology. He can be contacted at email: sandeepkampa5@gmail.com.
	**Srinivas Reddy Kosna** is a seasoned Software Developer and Quality Analyst with eight years of experience in AEM and Java/J2EE technologies. He specializes in developing AEM services, reusable components, templates, and workflows, with expertise in content migration and version upgrades. Proficient in AWS, dispatcher configurations, and Agile methodologies, he excels in web and enterprise application development. Srinivas has extensive knowledge of JCR, Apache Sling, and CQ5/AEM, along with hands-on experience in testing, version control, and cloud-based AEM infrastructure management. He can be reached at srinivas.k1290@gmail.com.