

Advancing Cybersecurity Education in the Era of Generative AI

Selina Demi ^[0000-0001-5988-4697], Bjørnar Guttormsen, Kevin Forberg Reinaas, Mats Hansen and Sofie Sand

Østfold University College, Halden, Norway

{selina.demi,bjornngu,kevinfr,math,sofiesa}@hiof.no

Abstract. Artificial intelligence advancements have recently received attention in both academic and industrial settings, demonstrating disruptive potential in many domains, including education. Previous studies have shown that AI-driven tools can transform conventional learning methods. This paper focuses on identifying how generative AI (GAI) tools are leveraged by cybersecurity students. A survey has been designed based on cognitive process dimension of Bloom's taxonomy and distributed to students studying cybersecurity in Norway. The results of the survey were illustrated with a set of use cases that demonstrated how Microsoft Copilot can enable students with skills ranging from lower-order knowledge-based skills to higher-order competence-based skills.

The applied use cases presented in this paper indicate that GAI tools can enhance lower- and higher-order thinking skills of cybersecurity students and contribute to bridging the gap between theoretical knowledge and competence-based skills needed to face real-world cybersecurity challenges. However, the results of the survey indicated that this potential has not yet been fully utilized by students, as they use such tools mainly for tasks that require lower-order thinking skills, such as remembering and understanding cybersecurity concepts, and less for more advanced cognitive tasks, such as creating role-playing scenarios. These preliminary findings can be useful to trigger discussions on how to improve future cybersecurity curricula to include applied GAI use cases and align them with learning outcomes. Ultimately, future efforts can be dedicated to developing a roadmap as a collaborative effort among policy-makers, cybersecurity academics and practitioners on the trustworthy incorporation of applied GAI in cybersecurity education.

Keywords: generative AI, cybersecurity education, Bloom's taxonomy, Microsoft Copilot, survey.

1 Introduction

Large language models (LLMs), such as generative pre-trained transformer (GPT) models have received substantial interest in both academia and industry [1]. The success of these advanced language models lies in their ability to handle a variety of applications, ranging from general natural language tasks to domain-specific tasks with

high performance [2]. Due to these capabilities, it is not surprising that users, such as students, with critical information needs are increasingly adopting generative artificial (GAI) systems that are built on such large language models [2]. According to previous research [3], GAI can transform conventional learning and teaching approaches and enable benefits for both students and teachers. Providing a personalized and interactive learning environment increases students' motivation and interest in learning which leads to significant improvement of learning outcomes and academic performance [3]. In addition, GAI has the potential to improve teaching skills and automate a set of repetitive teaching tasks, such as evaluation of students' assignments or providing feedback [3].

Despite these acknowledged benefits on education in general, there is very limited research on how cybersecurity education can leverage the recent advancements in GAI. It has been reported that current cybersecurity educational programs focus on traditional cybersecurity topics without adopting modern learning methodologies [4]. The interdisciplinary and ever-evolving nature of the cybersecurity landscape, coupled with academic settings of static nature have contributed to discrepancies between the knowledge acquired in academia and the skills required in industry [4]. In turn, this discrepancy has contributed to the growing cybersecurity skills shortage [4]. In this regard, the Computing Curricula 2020 (CC2020) report has demanded the transition from knowledge-based learning to competence-based learning [5]. To achieve this transition, educational institutions are encouraged to adopt innovative strategies to make the learning process engaging for students and to develop the necessary skills to prepare them for the realistic challenges that they will face as cybersecurity professionals. This paper aims to explore how GAI tools are currently used to advance cybersecurity education from the perspective of students.

The rest of the paper is structured, as follows: **Section 2** presents an overview of the current research on generative AI, evolution of LLMs, Bloom's taxonomy, cybersecurity education and generative AI in education. **Section 3** describes the research approach followed in this paper, comprising survey design and distribution, while **Section 4** presents the results of the survey and a set of use cases enabled by Microsoft Copilot with the potential to advance cybersecurity education. **Section 5** provides a reflection on the main results, discusses limitations and ethical considerations and presents a vision for GAI-driven cybersecurity education with promising future research directions. **Section 6** concludes the paper.

2 Background

2.1 Generative AI and evolution of Large Language Models

LLMs are advanced language models with the exceptional capability to learn patterns among unstructured texts by leveraging deep learning neural networks and use these patterns to generate contextually relevant responses based on text queries or prompts [6]. The fundamental features of LLMs that allow them to generate contextually relevant responses with improved performance over time are the self-attention module in

Transformer, context-learning, and Reinforcement Learning from Human Feedback (RLHF) [2]. Differently from previous single-task based AI applications, GAI chatbots based on LLMs enable interactive user experience by means of text-based dialogue [6]. One of the generative AI applications based on LLMs that has gained widespread popularity is ChatGPT built on GPT-3.5 and recently, GPT-4.

The first GPT model, i.e., GPT-1 was launched in 2018 with 115 million parameters and trained on the BookCorpus dataset [6]. Performance was improved in GPT-2 which was released in 2019 with 1.5 billion parameters and trained on WebText data consisting of over 8 million documents. Even higher performance was achieved with GPT-3 released in 2020. GPT-3 with 175 billion parameters was trained on 5 datasets (CommonCrawl, WebText2, Book1, Book2, Wikipedia) [6]. Fine-tuning of GPT-3 led to the development of GPT-3.5 that adopts a reward model trained on human feedback by ranking outputs produced by GPT-3.5. This reward model is referred to as RLHF. The accuracy of the responses was significantly improved with the release of GPT-4 in 2023, as a large multimodal model that accepts image and text inputs. Since the launch of ChatGPT built on GPT-3.5 and recently GPT-4, the interest on generative AI has mounted rapidly with the emergence of a variety of chatbots, such as Google's Bard, Baidu's Ernie, Microsoft's Bing Chat and Microsoft's Copilot [7].

Microsoft Copilot was adopted in the use cases presented in **Appendix**. Copilot uses a combination of different LLMs, such as GPT-4, but it can also be integrated with Microsoft Graph and Microsoft 365 apps, such as Word, PowerPoint, Outlook, Teams [8]. Copilot was chosen over other generative AI solutions, such as ChatGPT in this paper, due to two main factors: (i) Our preliminary comparison between different generative AI solutions indicated that Microsoft Copilot performs best when it comes to producing very detailed explanations of code in a variety of programming languages. In addition, Copilot provides relevant references and resources from the Web which is very important to enhance the learning experience of students. These findings are in line with previous studies, e.g., [9], (ii) integration with Microsoft Graph and Microsoft 365 apps make Copilot a powerful tool that can be used also in the academic setting to increase the productivity of both students and teachers.

2.2 Bloom's taxonomy

This paper adopts Bloom's taxonomy which refers to a classification of statements of what learning outcomes to expect from students, as a result of teaching [10]. In other terms, Bloom's taxonomy can be considered a framework for classifying the level of understanding that students acquire from what is being instructed [10]. The taxonomy consists of two main dimensions and their respective categories [10, 11], as follows: (i) knowledge dimension: factual knowledge, conceptual knowledge, procedural knowledge, metacognitive knowledge, (ii) cognitive process dimension: remember, understand, apply, analyze, evaluate, and create (See **Fig. 1**). The cognitive process dimension has been adopted in computer science education to develop and analyze programming assessments [12]. This paper leverages this dimension to structure the survey and to provide use cases that demonstrate how GAI can contribute to enhancing not only lower-order thinking skills of students, such as remembering and understanding,

but also higher-order skills, such as evaluating and creating. The use cases presented in this paper (See **Appendix**) align the cognitive process categories with the learning outcomes of the knowledge area of software security, as defined in the Cybersecurity Curricula 2017 (CSEC2017) report, published by the Association for Computing Machinery (ACM) [13].

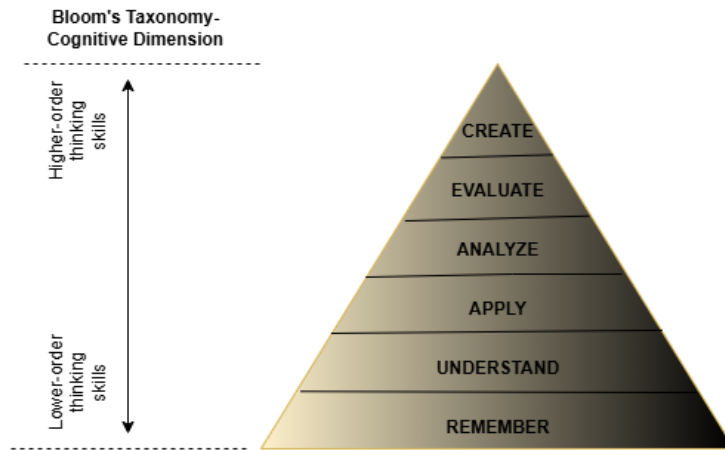


Fig. 1. Cognitive dimension in Bloom's Taxonomy, Adapted from [10].

2.3 Cybersecurity Education

The shortage of cybersecurity professionals has been acknowledged in literature [4, 14], and has been considered one of the strongest gaps in computer science [14]. The cybersecurity skills shortage has been attributed to two factors [14]: (i) high number of unfilled cybersecurity positions, and (ii) discrepancy between candidates' qualifications and skills needed in industry. The highly dynamic and ever-evolving cybersecurity field coupled with the academic setting of static nature has contributed to the latter discrepancy [4]. In fact, it is of utmost importance to update regularly the content of cybersecurity curricula to incorporate new emerging threats, how to address them, and other interdisciplinary skills needed to reflect the evolving nature of the cybersecurity field.

Furthermore, cybersecurity curricula should incorporate hands-on experience, in addition to theory-based knowledge, as lack of hands-on experience and training has been considered as one of the main concerns in cybersecurity education [4]. This means that there is a need to transition from knowledge-based learning to competence-based learning. This need was also highlighted in the Computing Curricula 2020 (CC2020) report [5]. To achieve this transition, educational institutions are encouraged to adopt innovative strategies to make the learning process engaging for students and to develop the necessary skills to prepare them for the realistic challenges that they will face as cybersecurity professionals. This paper aims to contribute to achieving this transition by exploring advanced and innovative technologies, such as GAI, that can shape the future cyber workforce.

2.4 Generative AI in education

AI technologies, such as chatbots have been used in education for decades, even though early chatbots lacked reliability [3]. The advent of deep learning technologies improved reliability and performance of chatbots, fostering their use in the education domain. As outlined in previous studies [3], implementing GAI technologies has the potential to provide personalized and interactive learning experiences, increasing motivation, engagement, and improving significantly academic performance. Interestingly, incorporating AI technologies into the learning experience can equip students with higher-order thinking skills, such as creativity and critical thinking [3], which are particularly needed in the cybersecurity field to deal with dynamic and complex cyber-attack scenarios.

Our preliminary literature review indicated only a few studies that focused on the topic of GAI in cybersecurity education [15], [16]. In [15], the authors presented a project for a cybersecurity course that allows students to take on cybersecurity roles in a specific industry, e.g., Chief Security Officer (CSO), interact with ChatGPT to identify attack vectors in a simulated cybersecurity scenario and plan strategic actions bridging in this way the gap between theory and practice. In [16], the authors presented a case study on the use of GAI-driven tools, such as ChatGPT to accelerate learning outcomes of a specific cybersecurity curriculum. Our paper adapts ideas from previous studies [15] [16], but with a different scope and methodology.

3 Research design

To explore how cybersecurity students currently use GAI tools in their studies, a structured questionnaire was designed and distributed to cybersecurity students at Østfold University College. The survey was designed based on Bloom's taxonomy learning framework [10], as it breaks down the process of learning in six comprehensive steps. The purpose was to capture students' practical applications of GAI in cybersecurity studies and categorize them based on different cognitive levels of student learning.

The survey consisted of two types of questions: closed-ended Likert-scale statements, and open-ended qualitative questions, in order to provide a more comprehensive understanding of students' experiences. Closed-ended Likert-scale statements allow for measurable and comparable data, making it easier to identify patterns and trends. On the other hand, open-ended questions offer deeper insights by giving participants the opportunity to explain their thoughts, share specific examples, and highlight perspectives that cannot be captured through fixed response options. Ultimately, combining both types of questions enhances the overall reliability and richness of the data collected.

3.1 Survey Design and Development

The survey is designed to be simple and easy to use by students. It consists of a set of sections that are explained in what follows:

Section I aims to gather basic background information about the participants and their general experience related to cybersecurity and GAI. The questions are delimited to areas, such as educational level, number of cybersecurity-related courses taken, and familiarity with AI tools. This delimitation was intentionally adopted to maintain the anonymity of the respondents while still providing relevant contextual data. **Section II** was designed to assess how generative AI tools support the first cognitive dimension in Bloom's taxonomy: *Remember*. This section explores how students use GAI to recall and review cybersecurity concepts, such as terminology, frameworks, and types of threats. **Section III** focuses on the second cognitive dimension in Bloom's taxonomy: *Understand*. The statements assess whether GAI helps students comprehend complex cybersecurity topics and explain them in simpler terms. **Section IV** focuses on the third cognitive dimension: *Apply*. This part explores the practical uses of GAI tools in cybersecurity assignments and projects. It includes tasks, such as scripting, writing reports, troubleshooting and problem solving.

Section V relates to the fourth cognitive dimension: *Analyze*. This section examines how students use GAI to break down cybersecurity problems, identify patterns, and evaluate threats, vulnerabilities, and the effectiveness of security measures. **Section VI** focuses on the fifth cognitive dimension: *Evaluate*. In this section, students reflect on whether GAI tools help them evaluate both their own work and existing material related to cybersecurity topics. It also encourages them to consider the reliability and trustworthiness of GAI-generated content. **Section VII** relates to the last cognitive dimension in Bloom's taxonomy: *Create*. This final section investigates whether students use GAI to generate ideas, design solutions, simulate cybersecurity scenarios, or create role-playing cybersecurity scenarios.

The six sections (**Sections II-VII**) are structured using Likert-scale statements to measure the extent to which students agree with various prompts related to their use of GAI tools. The Likert-scale provides a consistent and structured format for capturing student perceptions, ranging from Strongly Disagree to Strongly Agree. In each section, the statements are designed to align with specific learning behaviors, making it possible to assess how AI tools support students in recalling information, comprehending concepts, applying skills, analyzing problems, evaluating content, and generating new ideas. By combining the Likert scale with a taxonomy-based structure, the survey offers both clarity in design and depth in data collection.

In addition to the structured statements, several open-ended questions were included to give participants the opportunity to describe experiences that were not captured in the previous sections. These questions invited students to elaborate on: additional ways in which they use GAI in their studies, and specific situations where AI was particularly helpful. These questions were designed to be optional, in order to encourage participation and maximize the number of survey responses.

3.2 Survey Distribution

The survey is created using Nettskjema, a web-based survey tool developed by the University of Oslo. The platform allows users to create, store, and manage questionnaires securely. The platform is chosen for this study based on recommendations from

Østfold University College, where it is widely used for research and data collection purposes. The survey was originally written in English, but has also been translated to Norwegian, in order to lower the threshold for answering.

The survey was distributed using Østfold University's mailing list for IT-students, as well as posted in different servers and groups online. We allowed about a month for the answers to come in, before starting the analysis. The total sample size for this study consists of 30 respondents. One participant completed the original English version of the survey, while the remaining 29 respondents completed the translated Norwegian version. This approach ensured accessibility for Norwegian-speaking students while maintaining the option for English-speaking participants. The questions of the survey and the full detailed report of responses can be accessed online in a public repository [17]. This ensures transparency and reproducibility of our study.

4 Results

This chapter presents the analysis of the data collected through the survey on how cybersecurity students utilize GAI tools in their education. First, an overview of the respondent's demographics and general use of AI tools is provided. Following this, the responses to the Likert-scale statements are analyzed to identify patterns, trends and relationships between different aspects of GAI uses in cybersecurity education. Finally, the chapter explores the qualitative feedback provided by respondents in the open question section, offering deeper insights into their experiences, perceptions and challenges.

4.1 Background information

The survey data revealed that the majority of the respondents were pursuing Bachelor's degree in Cybersecurity or IT (23, 76,7%), while a smaller group were enrolled in Master's programs (7, 23,3%). None of the respondents indicated having a PhD, reflecting a focus on undergraduate and graduate-level students in the study. In terms of exposure to cybersecurity courses, the students displayed a diverse background, with 30% having completed one course, 23,3% two courses, 20% three courses, 3,3% four courses, and another 20% having completed five or more courses. Regarding their familiarity with generative AI tools, such as ChatGPT, most students reported having a moderate to high level of familiarity. Specifically, 26 of the 30 respondents indicated being either very familiar or extremely familiar with AI tools, while only 4 respondents described their familiarity as moderate. The frequency of AI tools usage among students varied slightly. 36,7% of the students reported using AI tools occasionally (a few times per month), while another 36,7% used them frequently (a few times per week). Notably, 20% of the students used AI tools daily, while only 6,7% reported using them rarely (a couple of times per semester).

4.2 Cognitive process dimension I: Remember

Remember was the first use case respondents were asked about in the survey, consisting of three statements. Among these statements, 19 students (63,3%) indicated that they use generative AI-tools to explore cybersecurity topics and concepts, 24 (80%) use them to summarize important cybersecurity frameworks and standards and 23 (76,7%) use them to review cybersecurity topics and knowledge before exams or assignments. Only 1 respondent strongly disagreed with each statement, while 3 participants disagreed.

The results indicate that GAI tools can be used to enhance lower-order thinking skills of cybersecurity students, such as remembering by retrieving relevant knowledge. To illustrate this cognitive dimension, a use case is presented in Appendix. In **Fig. 2**, Copilot is prompted about what buffer overflow is and why it is considered a security problem which is a learning outcome outlined in [13]. The tool provides a detailed response explaining buffer overflow, providing relevant references, and also historical buffer overflow attacks, such as Heartbleed (See **Fig. 2**).

4.3 Cognitive process dimension II: Understand

The second use case introduced was *Understand*. The use case consisted of four statements. 24 students (80%) indicated that they use GAI tools to help them understand cybersecurity assignments, 26 (86,6%) use the tools to understand cybersecurity concepts and definitions, 22 (73,3%) use them to understand cybersecurity frameworks and 23 (76,7%) use them to clarify cybersecurity related policies, regulations and best practices. Only 1 student reported that they strongly disagree with using these tools to clarify policies, regulations and best practices.

The results suggest that students use GAI tools to understand cybersecurity concepts and definitions, assignments, regulations and best practices. This is in line with the definition provided in [10]: “*Understanding information refers to determining the meaning of instructions, and encompasses cognitive process, such as explaining, interpreting, summarizing, inferring and classifying information*”. **Fig. 3** in **Appendix** illustrates this use case, by prompting Copilot to explain how secure code reviews are conducted, including the use of automated tools. Understanding these concepts is important to developing secure code, as outlined in [13].

4.4 Cognitive process dimension III: Apply

Use case three, *Apply* consisted of eight statements. 11 students (36,6%) indicated that they use GAI tools to structure their cybersecurity assignments, while 7 (23,3%) indicated that they do not use the tools for this purpose. The remaining 12 (40%) of responses to this statement were neutral. 10 students (33,3%) indicated that they use GAI tools to help write reports and documentation, while 13 (43,3%) indicated that they do not use the tools for this purpose. Moreover, 10 students (33,3%) revealed that they use GAI tools to simulate cyber scenarios, while 11 (36,6%) do not. Finally, 24

students (80%) reportedly use GAI tools for troubleshooting errors in practical security exercises.

The results indicate that GAI can be used by cybersecurity students to facilitate carrying out a procedure in a specific situation, advancing in this way the cognitive process category “*Apply*” [10]. **Fig. 4** in **Appendix** illustrates this use case, by prompting Copilot about the procedure of adding Content-Security-Policy to a HTML page using a tag, which is important to prevent CrossSite Scripting attacks.

4.5 Cognitive process dimension IV: Analyze

The fourth use case introduced is *Analyze*, which entailed four statements. 18 students (60%) indicated that they use GAI to help them analyze results from cybersecurity assignments. 17 students (56,6%) reportedly use GAI tools to help them analyze cybersecurity threats and vulnerabilities, with 10 students (33,3%) choosing to provide neutral response. Additionally, 21 students (70%) indicated that they use GAI tools to analyze strengths and weaknesses of different security strategies, and 18 (60%) use GAI to compare different cybersecurity tools or approaches.

These results indicate that cybersecurity students can use generative AI tools to analyze cybersecurity related information. A practical example of this use case is provided in Appendix, that prompts Copilot to analyze a code snippet for security vulnerabilities (See **Fig. 5**) and improve the code to address the identified vulnerabilities (See **Fig. 6**). Copilot provides a detailed response of security vulnerabilities and improvements of the code to address such vulnerabilities. This use case is useful for cybersecurity students to enhance the higher-order thinking skill “Analyze”.

4.6 Cognitive process dimension V: Evaluate

The fifth use case, referred to as *Evaluate*, consisted of four statements. 19 students (63,3%) revealed that they use GAI to evaluate their own work for cybersecurity assignments. 13 students (43,3%) indicated that GAI helps them think critically about cybersecurity issues, while 8 (26,6%) disagreed or strongly disagreed with the statement. In addition, 24 students (80%) reportedly compare AI-generated content with other cybersecurity sources before using it, and 26 (86,6%) indicated that they question AI recommendations before following them.

The results suggest that students use GAI to make judgments related to cybersecurity assignments. Evaluating is an important skill to develop, as cybersecurity professionals need to constantly make informed judgements, for instance regarding which security tools to integrate into the software development lifecycle. **Fig. 7** illustrates this use case by using Copilot to present a detailed comparative analysis of the most popular static analysis tools, including metrics and relevant web resources.

4.7 Cognitive process dimension VI: Create

The final use case introduced was *Create*, consisting of eleven statements. 11 students (36,6%) indicated that they use GAI tools to create secure code, 10 (33,3%) chose

to respond neutral and 9 (30%) disagreed or strongly disagreed with the statement. When asked if they use GAI tools to create cybersecurity role-playing scenarios, 8 (26,6%) agreed or strongly agreed, 7 (23,3%) responded with neutral and 15 disagreed or strongly disagreed. 15 students (50%) indicated that they do not use AI to create cybersecurity challenges to test their problem-solving skills or practice responding to cybersecurity incidents, 16 students also indicated that they do not use AI to role-play security policy decision making in an organization.

While GAI can equip students with higher-order thinking skills, such as critical thinking and creativity, as outlined in [3], the results indicate that students are not leveraging this potential. These are very important skills to prepare students to deal with an ever-changing and complex cyber threat landscape. **Fig. 8** illustrates how GAI can be leveraged to enhance the *Create* cognitive dimension. In this example, Copilot is prompted to develop cyber scenarios using role-playing techniques, such as simulating the role of Chief Information Security Officer (CISO). This exercise is indispensable for cybersecurity students, in order to understand the complexities of multifaceted roles, such as CISO through fictional case studies that simulate real-world scenarios.

4.8 Qualitative results

In the non-mandatory open-ended questions section, participants were invited to elaborate on their responses or add additional insights. Respondents stated that they mainly use AI tools in their cybersecurity studies to explain concepts and tasks related to coding. In the responses, there were four mentions related to explaining concepts and three mentions related to coding. Less prominent mentions include troubleshooting, exploiting vulnerabilities, and double checking. Moreover, broader variety of responses was observed when students were asked to elaborate on areas/situations they use GAI tools in their studies that were not previously mentioned. The responses ranged from how-to, planning, and templates to elaboration and translation. The most mentioned topic among these responses was rewording/rephrasing text, with a total of two mentions. When asked about specific cybersecurity topics or tasks where they find GAI particularly useful, the most mentioned topics included coding in six responses and explaining concepts in three responses. Other responses included a variety of cybersecurity topics, such as risk management, incident response, identity and access management (IAM), and threat detection. Additional topics, such as role-playing and analysis were also mentioned in the responses.

5 Discussion

5.1 Reflections on results

The preliminary results of this study indicated that cybersecurity students use GAI tools to a higher degree on use cases that enhance lower-order thinking skills, such as remembering and understanding. In fact, over 70% of students use GAI tools to perform a variety of tasks that relate to remembering and comprehending cybersecurity concepts

and definitions. This might suggest that students view GAI tools as a valuable learning aid when it comes to acquiring cybersecurity knowledge and comprehension of such knowledge. Use cases enhancing higher-order skills, such as applying and creating received higher levels of disagreement among the students, suggesting that they are less confident and more critical in GAI tools when it comes to more practical tasks. However, the results revealed high number of students that compare AI-generated content with other sources before using it (80%) and question AI recommendations before following them (86,6%). This suggests that students are developing critical evaluation skills regarding AI-generated content. Finally, it was observed that use cases that enhance higher-order skills, such as creating, were less common among cybersecurity students. While previous research [18] found out that the integration of GAI in cybersecurity enhanced significantly critical thinking skills of students, our results inferred that such potential has not yet been fully utilized by students.

5.2 Limitations and Ethical considerations

A key challenge encountered during this study involved obtaining a high number of survey respondents. The limited sample size may reduce the generalization of the findings and increase the impact of individual variations in responses. Additionally, while the survey was available in both English and Norwegian to accommodate a broader audience, only one respondent completed the English version, potentially limiting the diversity of perspectives captured. Finally, since the survey relies on self-reported data, there is a risk that some respondents may have exaggerated or underestimated how much they use GAI tools in their studies.

While the use of GAI tools can advance cybersecurity education, it also raises ethical concerns. Previous research [3] has pointed out concerns, such as biases in the training data leading to discriminatory responses, and privacy issues in terms of collecting personal data about students and teachers. In addition, these tools pose risk to academic integrity [3] [15] [19], for instance academic misconduct in essays or reports and cheating in exams. This is applicable to cybersecurity education, as mentioned in [19]. In addition, the growing use of GAI tools leads to the risk of these tools being fed with illegal, unethical and malicious content, such as conducting advanced phishing attacks and developing malware [19]. This increases the capabilities of such tools to be used in malicious ways. Given these ethical issues, it is important to be cautious when incorporating such tools in cybersecurity educational settings.

5.3 A vision of GAI-driven cybersecurity education

In what follows, we deliberate on a vision of GAI-driven cybersecurity education by discussing two promising future directions:

(i) Empirical evidence on the impact of GAI in addressing cybersecurity skills shortage

The lack of evidence on how GAI can impact cybersecurity students' motivation, interest in learning, critical thinking, academic productivity and performance indicates

the need for empirical studies in the field. Such studies can investigate how the incorporation of AI can address the gap between theory and practice to prepare cybersecurity students with the skills needed in industry, and consequently contribute to addressing the cybersecurity skills shortage. In this regard, our future work will be dedicated to conducting an empirical study that will consist of practical cybersecurity exercises with case studies from industry. Solving such exercises requires not only technical cybersecurity skills, but also critical thinking skills. Students will have to perform the exercises in two phases: (i) without the assistance of GAI tools, (ii) with the assistance of such tools. Surveys will be distributed to these students in both phases and their problem-solving, critical thinking skills, and performance will be measured. The analysis of the results will indicate the impact of GAI tools on enhancing higher-order thinking skills and performance of cybersecurity students.

(ii) Roadmap for the ethical use of GAI in cybersecurity education

The results of this paper suggest future cybersecurity curricula to incorporate applied GAI use cases and align them with learning objectives and outcomes. However, ad-hoc approaches for the use of GAI in cybersecurity education will not be able to fully leverage the potential of such innovative technologies. There is a need for a standardized, structured and uniform approach on the ethical integration of GAI tools into cybersecurity educational settings to bridge the gap between theoretical understanding and real-world application. Policy-makers, academics and cybersecurity practitioners need to develop collaboratively a roadmap on how to achieve this vision and leverage the benefits of GAI, while addressing ethical concerns.

6 Conclusion

The highly interdisciplinary nature of the cybersecurity profession and the fast-paced cyber threat landscape demand the transition from knowledge-based learning to competence-based learning. To achieve this transition, educational institutions are encouraged to adopt innovative strategies, such as GAI. This paper conducted an exploratory study to understand how students use GAI tools in their cybersecurity studies. A survey was designed and structured based on the six categories of the cognitive dimension of Bloom's taxonomy and distributed to cybersecurity students. The results of the survey were illustrated by practical use cases enabled by Microsoft Copilot, which could potentially be used as a starting point for development of specialized GAI tools for cybersecurity education.

The use cases indicate the broad applicability of GAI tools in enhancing lower- and higher-order thinking skills of cybersecurity students by serving as cybersecurity trainers, collaborators and facilitators in creating fictional cyber scenarios. However, the results revealed that this potential has not been fully utilized by cybersecurity students. According to the results, students use GAI tools mainly for tasks that require lower-order thinking skills, such as to remember and understand cybersecurity concepts and definitions, but less for more advanced cognitive tasks, such as creating cybersecurity role-playing scenarios. Therefore, there is work to be done to enhance future

cybersecurity curricula to include a variety of applied GAI use cases and align them with learning outcomes and objectives. Our research is a first step towards further discussions on how to effectively use GAI in educational settings to enhance students' learning outcomes. Future efforts need to be dedicated to developing a roadmap as a collaborative effort among relevant stakeholders, on the trustworthy incorporation of GAI in cybersecurity education.

Acknowledgments. We would like to thank all students who participated in this study.

References

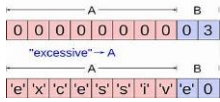
1. Hadi, M.U., Tashi, Q.A., Qureshi, R., Shah, A., Muneer, A., Irfan, M., Zafar, A., Shaikh, M.B., Akhtar, N., Wu, J., Mirjalili, S.: Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects, <https://www.techrxiv.org/doi/full/10.36227/techrxiv.23589741.v3>, (2023). <https://doi.org/10.36227/techrxiv.23589741.v3>.
2. Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., Ye, W., Zhang, Y., Chang, Y., Yu, P.S., Yang, Q., Xie, X.: A Survey on Evaluation of Large Language Models. *ACM Trans Intell Syst Technol.* 15, 39:1-39:45 (2024). <https://doi.org/10.1145/3641289>.
3. Adiguzel, T., Kaya, M.H., Cansu, F.K.: Revolutionizing education with AI: Exploring the transformative potential of ChatGPT. *Contemp. Educ. Technol.* 15, ep429 (2023). <https://doi.org/10.30935/cedtech/13152>.
4. Blažič, B.J.: Changing the landscape of cybersecurity education in the EU: Will the new approach produce the required cybersecurity skills? *Educ. Inf. Technol.* 27, 3011–3036 (2022). <https://doi.org/10.1007/s10639-021-10704-y>.
5. Cc2020 Task Force: Computing Curricula 2020: Paradigms for Global Computing Education. ACM, New York, NY, USA (2020). <https://doi.org/10.1145/3467967>.
6. Singhal, K., Azizi, S., Tu, T., Mahdavi, S.S., Wei, J., Chung, H.W., Scales, N., Tanwani, A., Cole-Lewis, H., Pfohl, S., Payne, P., Seneviratne, M., Gamble, P., Kelly, C., Babiker, A., Schärli, N., Chowdhery, A., Mansfield, P., Demner-Fushman, D., Agüera y Arcas, B., Webster, D., Corrado, G.S., Matias, Y., Chou, K., Gottweis, J., Tomasev, N., Liu, Y., Rajkomar, A., Barral, J., Sementurs, C., Karthikesalingam, A., Natarajan, V.: Large language models encode clinical knowledge. *Nature.* 620, 172–180 (2023). <https://doi.org/10.1038/s41586-023-06291-2>.
7. Rudolph, J., Tan, S., Tan, S.: War of the chatbots: Bard, Bing Chat, ChatGPT, Ernie and beyond. The new AI gold rush and its impact on higher education. *J. Appl. Learn. Teach.* 6, 364–389 (2023). <https://doi.org/10.37074/jalt.2023.6.1.23>.
8. camillepack: What is Microsoft 365 Copilot?, <https://learn.microsoft.com/en-us/copilot/microsoft-365/microsoft-365-copilot-overview>, last accessed 2025/08/02.
9. Agarwal, V., Garg, M.K., Dharmavaram, S., Kumar, D.: “Which LLM should I use?”: Evaluating LLMs for tasks performed by Undergraduate Computer Science Students, <http://arxiv.org/abs/2402.01687>, (2024). <https://doi.org/10.48550/arXiv.2402.01687>.

10. Krathwohl, D.R.: A Revision of Bloom's Taxonomy: An Overview. *Theory Pract.* 41, 212–218 (2002). https://doi.org/10.1207/s15430421tip4104_2.
11. Seaman, M.: Bloom's taxonomy. *Curric. Teach. Dialogue* Vol 13 1 2. 29 (2012).
12. Thompson, E., Luxton-Reilly, A., Whalley, J.L., Hu, M., Robbins, P.: Bloom's taxonomy for CS assessment. In: *Proceedings of the tenth conference on Australasian computing education-Volume 78*. pp. 155–161 (2008).
13. Joint Task Force on Cybersecurity Education: *Cybersecurity Curricula 2017: Curriculum Guidelines for Post-Secondary Degree Programs in Cybersecurity*. Association for Computing Machinery, New York, NY, USA (2018).
14. Goupil, F., Laskov, P., Pekaric, I., Felderer, M., Dürr, A., Thiesse, F.: Towards Understanding the Skill Gap in Cybersecurity. In: *Proceedings of the 27th ACM Conference on Innovation and Technology in Computer Science Education* Vol. 1. pp. 477–483. Association for Computing Machinery, New York, NY, USA (2022). <https://doi.org/10.1145/3502718.3524807>.
15. Mitra, R., Schwieger, D., Roy, I.: Educating the Next Generation of CSOs: An Exercise in Conversational Role Play with ChatGPT. (2023).
16. Jamieson, P., Bhunia, S., Rao, D.M.: With ChatGPT, Do We have to Rewrite Our Learning Objectives - CASE Study in Cybersecurity. In: *2023 IEEE Frontiers in Education Conference (FIE)*. pp. 1–5 (2023). <https://doi.org/10.1109/FIE58773.2023.10343052>.
17. Survey Results- Use Cases of Generative AI in Cybersecurity Education, https://figshare.com/articles/dataset/Survey_Results_-_Use_Cases_of_Generative_AI_in_Cybersecurity_Education/29815151/2, (2025). <https://doi.org/10.6084/m9.figshare.29815151.v2>.
18. Elkhodr, M., Gide, E.: Integrating Generative AI in Cybersecurity Education: Case Study Insights on Pedagogical Strategies, Critical Thinking, and Responsible AI Use, <http://arxiv.org/abs/2502.15357>, (2025). <https://doi.org/10.48550/arXiv.2502.15357>.
19. Al-Hawawreh, M., Aljuhani, A., Jararweh, Y.: Chatgpt for cybersecurity: practical applications, challenges, and future directions. *Clust. Comput.* 26, 3421–3436 (2023). <https://doi.org/10.1007/s10586-023-04124-5>.

Appendix

Prompt: What is a buffer overflow and why is it a potential security problem?

Response: A **buffer overflow** occurs when a program writes data to a buffer (a temporary storage area in memory) and exceeds the buffer's capacity. Imagine pouring 12 ounces of milk into an 8-ounce glass—it spills over and affects adjacent memory locations. In other words, too much information is passed into a container that lacks sufficient space, causing it to overwrite data in nearby containers¹.



Here's why buffer overflows are a **significant security problem**:

1. **Memory Corruption:**
 - When a buffer overflows, it overwrites adjacent memory areas. This can lead to unintended changes in program behavior, crashes, or unpredictable results.
 - Attackers exploit this behavior to manipulate memory contents deliberately.
2. **Code Execution Control:**
 - If an attacker can carefully craft input to overflow a buffer, they can overwrite memory locations containing executable code.
 - By replacing legitimate code with their own malicious instructions, attackers gain control over program execution.
3. **Exploitation Opportunities:**
 - Buffer overflow vulnerabilities allow attackers to:
 - Inject malicious code (e.g., shellcode) into memory.
 - Bypass security mechanisms (e.g., stack canaries, DEP/NX protection).
 - Escalate privileges or execute arbitrary commands.
4. **Historical Impact:**
 - The infamous **Heartbleed** vulnerability (2014) exposed millions of users due to a buffer overflow in SSL software.
 - Despite being well-understood, buffer overflow attacks continue to plague cybersecurity teams.

In summary, buffer overflows pose a serious risk to software security. Proper input validation, memory management, and secure coding practices are essential to prevent these vulnerabilities.

Learn more: 1. [cloudflare.com](https://www.cloudflare.com) 2. [csoonline.com](https://www.csoonline.com) 3. [fortinet.com](https://www.fortinet.com) 4. [owasp.org](https://www.owasp.org)

Fig. 2. Prompt and response for remembering buffer overflow vulnerability

Prompt: How are secure code reviews conducted?

Response: Conducting secure code reviews is crucial for identifying and addressing vulnerabilities in software applications. Here are **best practices** to ensure effective and thorough code reviews:

1. **Set Clear Objectives:**
 - Understand the vulnerabilities that are possible in the code.
 - Identify patterns of bad code you want to eradicate.
 - Review with a clear idea of what to look for.
2. **Review Only for Security:**
 - Focus specifically on security aspects during the review.
 - The more you look for, the less likely you'll find any vulnerabilities¹.
3. **Use Multiple Techniques:**
 - Combine complementary methods and tools:
 - **Manual Review:** A real person reviews the code line by line.
 - **Automated Secure Code Review Tools:** These tools scan the code and report flaws.
 - Each method reveals different issues, improving the overall results².
4. **Review Regularly:**
 - Consistently review code, especially when meaningful changes are introduced.
 - Regular reviews lead to more secure and higher-quality code.
 - Consider using automated code review tools for quick and efficient checks³.
5. **Validate Input and Output:**
 - Pay attention to how input is handled (e.g., user input, API calls).
 - Ensure proper validation, escaping, and sanitization to prevent injection attacks (e.g., SQL injection, XSS).
6. **Understand the Big Picture:**
 - Look beyond individual lines of code.
 - Consider the overall architecture, data flow, and system interactions.
 - Understand the intent of the review and how components fit together¹.

Remember that secure code reviews are an ongoing process. Regularly assess your codebase, involve team members, and stay informed about emerging threats.

Learn more: 1. blog.securityinnovation.com 2. threatintelligence.com 3. codegrip.tech

Fig. 3. Prompt and response for understanding secure code reviews

Prompt: How can Content-Security-Policy be added to a HTML page using a <meta> tag to prevent Cross-Site Scripting attacks?

Response: Certainly! To add a **Content-Security-Policy (CSP)** to an HTML page using a <meta> tag, follow these steps:

1. **Meta Tag Approach:**

- The <meta> tag allows you to define a CSP policy directly within your HTML document.
- While this approach is less powerful than using an HTTP response header, it can be useful when you can't set headers directly (e.g., in certain hosting environments).
- Here's an example of how to add a CSP policy using a <meta> tag:

```
<meta http-equiv="Content-Security-Policy" content="default-src 'self'; img-src https://*; child-src 'none';">
```
- In this example:
 - default-src 'self' allows loading resources from the same origin (the current domain).
 - img-src https://* permits loading images from secure (HTTPS) sources.
 - child-src 'none' disallows loading content in frames or iframes.
- Place the <meta> tag inside the <head> section of your HTML document.
- Note that the CSP policy only applies to content found after the meta tag is processed, so keep it towards the top of your document or before any dynamically generated content.

2. **Considerations:**

- **HTTP Header Approach (Recommended):**
 - If possible, configure your web server to return the Content-Security-Policy HTTP header.
 - This header provides stronger security and additional features (e.g., reporting).
 - An HTTP header-defined CSP is more effective against attacks.
- **Meta Tag Limitations:**
 - Some directives (e.g., frame-ancestors, sandbox) are not supported inside a meta tag.
 - The Content-Security-Policy-Report-Only header cannot be used in a meta tag.
 - For ultimate protection, use HTTP response headers whenever feasible.

3. **Why Use CSP?:**

- CSP helps prevent **Cross-Site Scripting (XSS)** attacks by controlling script execution and resource loading.
- Define valid sources for scripts, styles, images, and other resources.
- Specify allowed domains and protocols (e.g., enforce HTTPS).
- Mark cookies with the secure attribute.
- Use the Strict-Transport-Security header for encrypted connections.

Remember that while the <meta> tag approach is handy when you can't set HTTP response headers directly, using an HTTP response header is generally a stronger security practice.

Fig. 4. Prompt and response for applying content-security-policy

Prompt: Analyze the following JAVA code:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class ProductSearch {
    public static void main(String[] args) {
        String userInput = "105 OR 1=1"; // User input (simulating an attacker)

        try {
            // Connect to the database (replace with your actual database connection details)
            Connection connection = DriverManager.getConnection("jdbc:mysql://localhost/mydb", "username", "password");
            Statement statement = connection.createStatement();

            // Vulnerable query: directly concatenates user input
            String query = "SELECT * FROM products WHERE category = '" + userInput + "'";
            ResultSet resultSet = statement.executeQuery(query);

            // Process the results (e.g., display product information)
            while (resultSet.next()) {
                int productId = resultSet.getInt("product_id");
                String productName = resultSet.getString("product_name");
                System.out.println("Product ID: " + productId + ", Name: " + productName);
            }

            // Close resources
            resultSet.close();
            statement.close();
            connection.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Fig. 5. Prompt and response for analyzing JAVA code for security vulnerabilities (Part I)

Response: Here's an improved version of the code that prevents SQL injection:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
public class ProductSearchSecure {
    public static void main(String[] args) {
        String userInput = "105 OR 1=1"; // User input (simulating an attacker)
        try {
            // Connect to the database (replace with your actual database connection details)
            Connection connection = DriverManager.getConnection("jdbc:mysql://localhost/mydb",
                "username", "password");
            // Use a parameterized query to prevent SQL injection
            String query = "SELECT * FROM products WHERE category = ?";
            PreparedStatement preparedStatement = connection.prepareStatement(query);
            preparedStatement.setString(1, userInput);
            ResultSet resultSet = preparedStatement.executeQuery();
            // Process the results (e.g., display product information)
            while (resultSet.next()) {
                int productId = resultSet.getInt("product_id");
                String productName = resultSet.getString("product_name");
                System.out.println("Product ID: " + productId + ", Name: " + productName);
            }
            // Close resources
            resultSet.close();
            preparedStatement.close();
            connection.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Improvements:

- We replaced the vulnerable query string with a parameterized query using ? as a placeholder.
- The actual value of userInput is passed separately, preventing SQL injection.
- Parameterized queries automatically handle escaping and sanitization.

Fig. 6. Response for addressing security vulnerabilities in JAVA code snippet (Part II)

Prompt: Provide a comparative evaluation of the most popular static analysis tools.

Response: Certainly! Let's compare some popular static code analysis tools based on their features, language support, accuracy, and other factors. Here's a comparative analysis:

1. **SonarQube:**
 - **Type:** Open-source static code analysis tool.
 - **Purpose:** Identifies errors, security issues, and code smells.
 - **Features:**
 - Continuous inspection of code quality.
 - Automatic reviews.
 - Supports various languages.
 - **Availability:** Free and paid versions.
 - **Platforms:** Docker (Windows, Linux, macOS, and Azure)¹.
2. **Checkmarx SAST:**
 - **Type:** Enterprise-grade static analysis tool.
 - **Purpose:** Identifies security vulnerabilities early in development.
 - **Features:**
 - Flexible and accurate.
 - Integrates well into development environments.
 - **Use Case:** Security-focused code scanning.
3. **Veracode Static Analysis:**
 - **Type:** Static code analysis tool.
 - **Purpose:** Scans deployments before release.
 - **Features:**
 - Automated feedback and guidance.
 - Cuts down mistakes.
 - Small digital footprint.
4. **Snyk Code:**
 - **Type:** Quick and effective static code analysis tool.
 - **Purpose:** Finds bugs and vulnerabilities.
 - **Features:**
 - High scan speeds.
 - Uses semantic analysis.
 - Free for individual developers and small teams.

Remember that the choice of tool depends on your specific requirements, project size, and team preferences. Consider factors like ease of integration, language support, and accuracy when selecting a static analysis tool.

For a more detailed comparison, you can explore additional resources such as the Top 5 Static Code Analysis Tools in 2023 and the Compare 600+ Linters, Static Analysis Tools, and Code Formatters^{2 3}.

Learn more: 1. [comparitech.com](https://www.comparitech.com) 2. [expertbeacon.com](https://www.expertbeacon.com) 3. [analysis-tools.dev](https://www.analysis-tools.dev)

Fig. 7. Prompt and response for evaluating static analysis tools

Prompt: Create a fictional case study that requires secure software development assessment. Provide the code snippet and identify the attack surface. Act as the CISO of the organization and take remedial actions.

Response: Certainly! Let's create a fictional case study involving a web application for an e-commerce platform called "ShopSafe." As the Chief Information Security Officer (CISO) of ShopSafe, I'll assess the security of the provided code snippet and propose remedial actions.

Case Study: ShopSafe E-Commerce Platform

Scenario:

ShopSafe is an online marketplace where users can buy and sell products. The platform allows users to create accounts, browse products, add items to their cart, and make purchases.

Code Snippet (Vulnerable Implementation):

Below is a simplified snippet of the user registration functionality in ShopSafe:

```
# shop_safe.py
import sqlite3
def register_user(username, password):
    try:
        conn = sqlite3.connect("shopsafe.db")
        cursor = conn.cursor()
        # Vulnerable query: directly concatenates user input
        query = f"INSERT INTO users (username, password) VALUES ('{username}', '{password}')"
        cursor.execute(query)
        conn.commit()
        print("User registered successfully!")
    except sqlite3.Error as e:
        print(f"Error: {e}")
    finally:
        conn.close()
if __name__ == "__main__":
    username = input("Enter your username: ")
    password = input("Enter your password: ")
    register_user(username, password)
```

Attack Surface:

SQL Injection:

- The register_user function directly concatenates user input (username and password) into the SQL query.
- An attacker can manipulate the input to execute arbitrary SQL commands.

Remedial Actions:

1. **Parameterized Queries (Prepared Statements):**
 - Rewrite the query using parameterized queries to prevent SQL injection.
 - Use placeholders (?) and bind user input separately.
2. **Hash and Salt Passwords:**
 - Store hashed and salted passwords in the database.
 - Use a strong cryptographic hash function (e.g., bcrypt) to protect user credentials.
3. **Input Validation:**
 - Validate user input (e.g., check for valid usernames, strong passwords).
 - Sanitize input to prevent special characters.
4. **Database Security:**
 - Restrict database permissions to minimize attack surface.
 - Regularly audit and monitor database activity.
5. **Security Code Review:**
 - Conduct regular code reviews to identify and fix security issues.

Secure Implementation (Sample):

```

# shop_safe_secure.py
import sqlite3
import hashlib
import secrets

def hash_password(password):
    salt = secrets.token_hex(16)
    hashed_password = hashlib.pbkdf2_hmac("sha256", password.encode(), salt.encode(), 100000)
    return salt, hashed_password

def register_user_secure(username, password):
    try:
        conn = sqlite3.connect("shopsafe_secure.db")
        cursor = conn.cursor()
        salt, hashed_password = hash_password(password)
        query = "INSERT INTO users (username, salt, hashed_password) VALUES (?, ?, ?)"
        cursor.execute(query, (username, salt, hashed_password))
        conn.commit()
        print("User registered securely!")
    except sqlite3.Error as e:
        print(f"Error: {e}")
    finally:
        conn.close()

if __name__ == "__main__":
    username = input("Enter your username: ")
    password = input("Enter your password: ")
    register_user_secure(username, password)

```

As the CISO, I recommend implementing the secure version (shop_safe_secure.py) and conducting thorough security testing. Regularly assess the entire application for vulnerabilities and stay informed about emerging threats.

Fig. 8. Prompt and response for creating a fictional cyber scenario and using the role-playing technique