An Extended Review: LLM Prompt Engineering in Cyber Defense

Neethu Shenoy
School of Technology and Innovation
Marymount University
Arlington, USA
n0s57100@marymount.edu

Alex V Mbaziira
School of Technology and Innovation
Marymount University
Arlington, USA
ambaziir@marymount.edu

2024 International Conference on Electrical, Computer and Energy Technologies (ICECET) | 979-8-3503-9591-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICECET61485.2024.10698605

***Abstract*— The launch of ChatGPT in November 2022 marked a significant advancement in the field of artificial intelligence, particularly in the realm of generative AI (GAI). ChatGPT is based on the Large Language Model (LLM). Use of AI in Cybersecurity can prove to be beneficial as the security analysts are employing AI models for improved detection of threats and quicker response to incidents. The interaction with LLMs needs to be skillfully and tactfully created to get precise and concise response. This technique of crafting queries to interact with the LLM is called prompt engineering. Prompt engineering in vulnerability detection and management is a new trend in the industry to manage cybersecurity threats and weaknesses proactively and effectively. This paper presents an extensive review of the field of prompt engineering in cybersecurity. It is primarily based on a comprehensive analysis of existing literature, encompassing a wide range of sources to provide a thorough overview of the current state and advancements in AI. The review delves into various aspects of prompt engineering, synthesizing key findings and theories from a multitude of scholarly articles and industry reports. This approach ensures a holistic understanding of the AI models including LLMs and how generative AI, in terms of prompt engineering, can be used for cyber-defense.**

Keywords: Artificial Intelligence, Machine Learning, Natural Language Processing, Generative AI, Large Language Models, Prompt Engineering, ChatGPT, Cyber-Defense, Cybersecurity, Vulnerability Management

I. Large Language Model

Large Language Models (LLM) are deep learning algorithms that are designed to understand, generate, and work with human language. These models are trained on massive amounts of text data to learn a wide variety of language patterns, meanings, and nuances. LLMs can read and comprehend text. They can grasp the meaning of sentences, questions, and even complex paragraphs. LLMs can be used to write and generate text such as answering a question, writing a story, or even composing an email. The versatile use of LLMs applies widely in many areas such as chatbots, translation services, content creation tools, and more. In summary, an LLM is an advanced and intelligent system that can read and write in human language, learning from a vast amount of text data to be as accurate and helpful as possible.

LLMs are based on transformer networks that is a type of machine learning architecture for handling sequential data like text or time series. This has revolutionized the field of Natural Language Processing (NLP). Transformers are important in handling text sequences and capturing the context around each piece of text appropriately. Transformers gained popularity in a short time-frame due to their capability to scale easily. They can be made larger and trained on vast amounts of data, which improves their performance.

A transformer consists of several components called transformer blocks or layers. These include self-attention layers, feed-forward layers, and normalization layers, which all collaborate to interpret incoming data and generate predictions for the output during the inference process. By stacking these layers, it is possible to create deeper and more potent transformers, resulting in powerful language models. Transformer networks are a powerful and efficient way to process and understand sequential data, especially language, and they have significantly advanced the capabilities of AI in understanding and generating human-like text.

The NLP algorithms garnered more attention with the introduction of transformer-based models like BERT ((Bidirectional Encoder Representations from Transformers), GPT (Generative Pre-trained Transformer), Transformer-XL, XLNet, RoBERTa (Robustly Optimized BERT Approach) and T5 (Text-To-Text Transfer Transformer). All the above models have been developed to carry out tasks like sentiment analysis, question answering, understanding, and interpreting long documents, and improve versatility.

The development of LLMs has given rise to new areas of study focused on how to effectively retrieve information from these models, known as prompt engineering. Researchers have investigated and utilized various techniques for prompt engineering, including few-shot prompting, chain-of-thought reasoning, and self-consistency methods.

II. Prompt Engineering

Prompt engineering is the science of prompting for inputs to interact with LLM. It is the technique of skillfully designing and formulating prompts or input queries to interact effectively with artificial intelligence models, especially those based on natural language processing. The aim of prompt engineering is to produce the most accurate, relevant, and useful responses from the AI. Prompt Engineering involves:

- Selecting the right language usage to accurately convey the meaning and intent of the question.

979-8-3503-9591-4/24/$31.00 ©2024 IEEE

- Providing sufficient context to the question to obtain on-topic and comprehensive response from the AI model.
- Understanding the models that are being used for prompting to utilize its full potential.
- Crafting the questions in a way that does not create confusion or minimizes the chances of misinterpretation by the model.
- Being creative in querying to experiment different capabilities of the model and obtain varied responses.

Most LLM models are adept at comprehending natural human language due to their training on extensive literary content. These models can be queried directly without needing a specific format. [1] have elucidated the workings of prompt engineering. For instance, when posed with a query like "Is Iris a flower?", the model perceives it as a classification task. It embarks on understanding the characteristics of a flower and its various types. Subsequently, it assesses the connection between Iris and flowers. After processing this information, the model formulates an answer and communicates it back to the user in natural, human-like language. Before going deeper into the concepts of prompt engineering, it is essential to know the fundamental rules of creating prompts. They are as follows:

- Keep it simple: the prompting question or text must be simple - possibly a single line and well-defined ask. This simplicity will fetch accurate and useful responses from the LLM model.
- Be specific: The instructions provided to the model must be specific and explicit. This will help the model to elicit precise and applicable responses.
- Be concise: the prompts must be kept concise and to the topic. This ensures the interaction is not complicated, providing sufficient clarity about the topic.

Prompting is based on four data quadrants as shown in the figure below. The prompt question is: “In the wake of software supply chain attacks like SolarWinds, Colonial Pipeline attack, and Kaseya ransomware, explain the Software Bill of Material (SBOM) using the National Telecommunications and Information Administration (NTIA) report released in July 2021 in simple text no more than two pages.”.

Context provides the background and helps the model develop a response based on the scenario stated in the context. Instruction tells the model what it is expected to do and what actions to perform. Input text is the information piece in the form of a document, text, numbers, or any other data point, that the model must reference while generating the response. The output pointer indicates the format or style the model must output the response.

Prompts in LLM models are not limited to straightforward queries. [1] specifies in their study that after the pre-training phase, they can perform a range of natural language processing tasks, including summarization, sentiment analysis, paraphrasing, handling open-ended questions, and detecting similarities. These functions are pivotal to mimicking human comprehension and are often employed to evaluate LLMs' performance. Furthermore, the reasoning or "thought" process of an LLM can be examined, leading to a comprehensive Chain of Thought (CoT) analysis, which is useful for identifying the model's knowledge shortfalls.

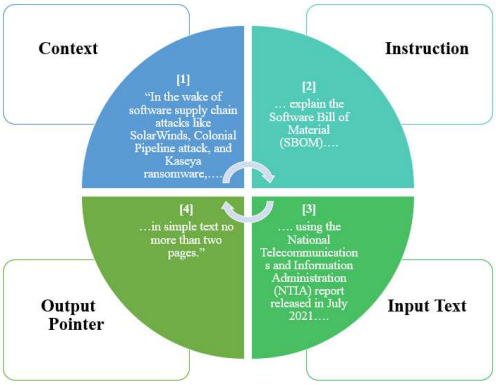

Fig. 1. The four quadrants of Prompting

The model’s response can vary depending on the prompt, particularly if additional information or context is required. [1] has cited some examples in their paper – for example, a model trained with data up to 2021 won't have accurate information about events occurring in 2022. In such scenarios, essential data not known to the model can be included in the prompt, serving as a source of new information for the model. This approach is expandable for teaching new tasks to the model through a concept known as ICL (In-Context Learning).

The LLM models learn from the examples and vast amounts of data fed to the model. There are different training methods, such as zero-shot learning, one-shot learning, and few-shot learning. In zero-shot learning, the LLM generates response solely on its fundamental training, without any specific examples. One-shot learning uses an example to generate context-based output. This could be a build up from zero-shot learning. In few-shot learning, the model is presented with multiple examples to fine-tune its response and provide precise output.

[1] dives deep into the approaches to optimize the accuracy of the prompts using techniques like BatchPrompting and MultiStagePrompting. These are advanced strategies to maximize the potential of LLMs in complex tasks. BatchPrompting involves sending a batch of prompts or queries to an LLM model at once, rather than one at a time. The advantage of BatchPrompting is efficiency. By processing multiple prompts simultaneously, the model can leverage its parallel processing capabilities, which is particularly useful for large-scale data analysis or when time is a critical factor. Additionally, BatchPrompting can sometimes improve the consistency and contextuality of the responses, as the model can use the information from multiple prompts to better understand the overall context. MultiStagePrompting is a more complex approach where the output of one prompt is used as the input for the next prompt, in a sequential or staged manner. This method is useful for tasks that require multiple steps of reasoning or when the output of one query influences the next. For example, the first prompt might ask the model to read a text and summarize it, and the subsequent prompt could ask it to analyze the sentiment of that summary. MultiStagePrompting is akin to having a conversation with the model, where each response informs the next question.

Kartal et.al highlighted a couple concepts in Prompt Engineering such as Chain of Thought (CoT) and In-Context Learning (ICL). Introduced by [2], the CoT method empowers AI systems to engage in intricate reasoning processes by breaking down complex tasks into smaller, intermediate steps. This approach becomes especially advantageous when combined with the few-shot prompting technique, as it leads to enhanced performance on challenging tasks that necessitate comprehensive reasoning abilities prior to producing accurate responses. Few-shot prompting is a technique to teach the models to perform tasks with very limited examples or "shots" of training data. This approach is particularly useful in the absence of large volumes of labeled data for a specific task. In-Context Learning (ICL) refers to a process in which a language model is fine-tuned or updated with additional knowledge and information while it is actively deployed and interacting with users or a specific environment. This approach allows the model to adapt and improve its performance over time based on the context in which it is being used. Kartel et.al in their paper explains that due to the capacity for comprehensive understanding, the models could acquire knowledge for new tasks on-the-fly through prompts during the inference process. Based on the context within the prompt, the model could gather the question and address or find similar instances. The sequential querying by the user - initially describing the intended task with examples; and subsequently, including the specific task in the prompt, provides sufficient contextual information to the model within the prompt. Depending on the response, the user can employ an iterative refinement process to offer improved context and send a fresh query to the model, repeating this process until the desired outcome is achieved. In summary, CoT and ICL are both techniques used in the realm of LLMs and NLP, but they serve somewhat different purposes and have distinct characteristics. CoT focuses on maintaining conversation coherence and relevance during multi-turn interactions, while ICL focuses on enabling the model to learn and adapt to new tasks or domains during real-time interactions. They can complement each other in applications where both conversational quality and adaptability are important.

Prompt engineering has gained immense popularity with different generative AI models. The technique is used in prompting textual responses such as in ChatGPT. Midjourney is another generative AI program that produces images from natural language descriptions. This program uses prompts from the user to learn the characteristics and generate images that match the explanations. The GitHub Copilot is an AI model that helps in generating code faster. Adequate prompting helps the Copilot to understand what is being asked. The quality of the response depends on how clear the instructions are.

III. Generative AI in Cyber Defense

With the latest breakthroughs in Artificial Intelligence and Machine Learning, scientists and researchers are now shifting their attention toward reshaping the cybersecurity landscape. LLMs are proving to be valuable tools in the field of cyber-defense by offering various capabilities and applications that enhance cybersecurity efforts. Below are some ways the LLMs are used in securing the information systems from cyber-attacks:

- Threat Intelligence and Analysis: - LLMs can be used to analyze vast amounts of textual data from sources like forums, blogs, news articles, and social media to identify potential threats and vulnerabilities. They can assist in natural language processing tasks like sentiment analysis, trend detection, and entity recognition to uncover emerging threats and patterns.
- Vulnerability Management: - LLMs can assist in scanning and analyzing codebases to identify potential security vulnerabilities and offer recommendations for remediation.
- Minimize false positives: - A significant issue revolves around the substantial number of false alarms triggered by security systems. LLM models can play a role in mitigating false positives by enhancing the queries employed for communication, leading to more accurate detection of threats, alleviating the workload on security teams who would otherwise need to investigate incidents that do not pose actual risks.
- Incident Response: - During a security incident, LLMs can assist in generating incident reports, helping analysts document and track the incident response process effectively.
- Security Policy and Compliance: - LLMs can assist organizations in understanding and adhering to security policies and compliance standards by providing detailed explanations and guidance.
- User Awareness and Training: - LLMs can be used to create interactive and personalized security awareness and training materials for employees, helping them recognize and respond to potential threats.

There are many research papers and studies published in the last couple of years that explain the advancements in LLM-supported cyber defense and adversarial landscape. The study discusses the importance of identifying vulnerabilities to enhance security defensive resilience and introduces the idea of integrating AI techniques, particularly LLMs, into threat exercises. Here the emphasis is on identifying security vulnerabilities that may not be addressed by existing security measures. Red teams and penetration testers assist in this process by conducting realistic, yet controlled threat simulations aimed at uncovering vulnerabilities that could be exploited by malicious actors. Penetration testers discover these vulnerabilities, enabling system administrators to patch and strengthen their network defenses.

The most extensive and high-performing LLM models undergo training with datasets that consist of several million words. These datasets are constructed by collecting text from a variety of sources on the internet - web content, articles, research papers, and books. Given the immense scale of these datasets, it is reasonable to assume that LLMs have undergone training that incorporates cybersecurity-related reports and reference materials. This training likely encompasses both offensive elements, such as threat-related information, and defensive components, such as security-related code. It is probable that these models have assimilated a wide range of publicly accessible cyber information and data on cyber frameworks, including enumerations of

vulnerabilities and weaknesses documented in Common Weakness Enumeration (CWE), Common Vulnerabilities and Exposures (CVE) databases, publicly available data on Advanced Persistent Threats (APTs) as per the MITRE ATT&CK framework, attack patterns detailed in the Common Attack Pattern Enumeration and Classification (CAPEC), documented exploits found in exploitDB, as well as strategies and tools associated with penetration testing, often found in online guides.

The study conducted by [3] in the paper "LLMs Killed the Script Kiddie: How Agents Supported by Large Language Models Change the Landscape of Network Threat Testing" explores the possibility of engineering the LLM prompts to assist the penetration testers to allow easier access to vulnerability scans and security audits. The researchers use prompt chaining which is a prompt engineering technique to produce consistent and executable actions. Crafting prompts for language models is something of an art form, and researchers are continuously exploring the most effective techniques for eliciting desired information. To gain insight into the specific impact of each component within a prompt, the experiments were conducted by deconstructing the Execution Stage prompt sentence-wise and observing how the language model's response varied as a result. The language model tends to perform well when provided with clear and explicit commands, especially when generating responses that remain unaffected by grammatical or spelling errors. The researchers in this study claimed that with prompt engineering "…much of the intelligence of the human threat actor effort devoted to 'figuring out what to do' during a campaign has transformed to 'figuring out how to tell a LLM to figure out what to do'…". The authors emphasize that prompt engineering requires acumen and skill that needs to be developed for advanced threat analysis.

IV. Prompt Engineering for Vulnerability Management

As seen from the above sections, prompt engineering involves the careful formulation and refinement of questions or commands to generate targeted, beneficial responses from AI models. This methodical approach aligns the capabilities of generative AI with human goals and organizational requirements, producing responses that meet specific objectives. In the previous section, it is elaborated that LLMs can be used extensively in the field of cybersecurity. This paper highlights how prompt engineering can be used in vulnerability management to detect, triage, prioritize, and remediate vulnerabilities.

Cybersecurity vulnerabilities refer to weaknesses in a system or network that can be exploited by malicious users to gain unauthorized access into the system, cause damage, or steal data or information. Understanding the various types of vulnerabilities in the first step in effective cyber defense. By crafting specific prompts, security professionals can guide the AI tools to focus on particular aspects of the code. This means they can tailor the analysis to look for common vulnerabilities, such as buffer overflows, SQL injections, or cross-site scripting (XSS) flaws, thereby making the detection process more efficient.

The diagram below illustrates the block diagram of an LLM model and the data flow when a command is injected by a user or a cybersecurity professional. The process begins with the query or question being submitted to the application (e.g., ChatGPT), where it is embedded. These embeddings are stored in a vector database and are searched against the user query. The results are then sent back to the application. The LLM model processes the user query using the trained model to compute the match, and the final output is returned to the user through the application.

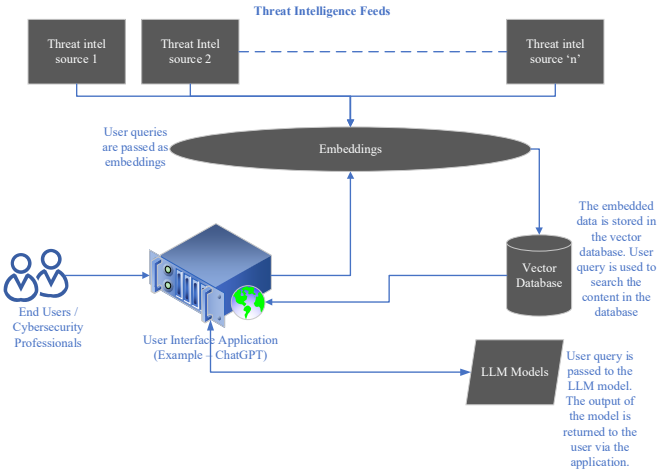

Fig. 2.Block Diagram of the LLM Model

Below are a few types of vulnerabilities and examples to demonstrate how prompt engineering helps in detecting these vulnerabilities.

Software Code Weaknesses: Flaws or weaknesses in software applications or operating systems that can be exploited. Examples include buffer overflows, injection flaws (like SQL injection), and cross-site scripting (XSS).

1. Buffer Overflow Detection:
   a. Prompt: "Analyze this C++ code and identify any instances where buffer overflow might occur. Look for unsafe functions like strcpy, sprintf, and arrays that don't check boundaries."
   b. Action: This prompt directs the AI tool to scan C++ code for classic buffer overflow vulnerabilities, particularly focusing on functions known to be risky without proper handling.
2. SQL Injection Vulnerability Scan:
   a. Prompt: "Review this PHP code for potential SQL injection points. Highlight code segments where user input is used in SQL queries without proper sanitization or parameterized queries."
   b. Action: This instructs the AI to pinpoint areas in PHP code where user input might be incorporated into SQL queries, a common source of SQL injection vulnerabilities.
3. Cross-Site Scripting (XSS) Check:
   a. Prompt: "Examine this JavaScript and HTML code for XSS vulnerabilities. Look for outputs where user input is directly reflected without adequate escaping or validation."
   b. Action: This prompt helps focus the AI's attention on parts of the code where user input is outputted, potentially leading to XSS attacks if not properly handled.

Infrastructure Vulnerabilities: These are defined as weaknesses that emerge in a computer's firmware or operating system, often due to the use of outdated software or software that has reached its end-of-life phase. Such vulnerabilities can also manifest in network devices, including routers and switches. These issues can typically be remediated through patches and updates published by the software vendor.

1. Operating System Vulnerability Scan:
    a. Prompt: "Perform a deep scan on this operating system version for known vulnerabilities listed in the latest CVE database that have not been patched in the current system update."
    b. Action: This prompt directs the AI to cross-reference the operating system's version against a database of known vulnerabilities (CVE database) to identify any unpatched issues.
2. Network Infrastructure Check:
    a. Prompt: "Analyze the firmware and configuration settings of these network routers and switches for any unpatched vulnerabilities or non-compliance with the latest security protocols."
    b. Action: This prompt would guide the AI to investigate both the firmware version and the configuration settings of network devices to spot unpatched vulnerabilities.

Network Vulnerabilities: Weaknesses in network architecture or configuration. Examples include unsecured wireless networks, outdated protocols, or poorly configured firewalls.

1. Network Configuration Analysis:
    a. Prompt: "Examine this network's configuration for common misconfigurations or weak points, such as open ports, default passwords, or unsecured wireless access points."
    b. Action: This prompt directs the AI to scrutinize the network's setup, looking for easily overlooked but common vulnerabilities.
2. Firewall Rules and Policies Audit:
    a. Prompt: "Review and analyze the firewall's current rules and policies for any loopholes or overly permissive settings that could allow unauthorized access or data leaks."
    b. Action: This involves instructing the AI to assess the firewall configuration for potential vulnerabilities or ineffective rules.

Physical Vulnerabilities: Related to the physical security of IT assets. Examples include lack of secure access to data centers or server rooms, and insufficient protection against environmental hazards.

1. Data Center Physical Security Check:
    a. Prompt: "Review the physical security measures of this data center, including access controls, surveillance systems, and fire suppression methods, for any vulnerabilities or non-compliance with industry standards."
    b. Action: The AI is tasked with evaluating the adequacy of physical security measures in a data center, identifying potential vulnerabilities in its protection systems.
2. Industrial Control Systems Security:
    a. Prompt: "Evaluate the physical security measures in place for these industrial control systems, focusing on access to control rooms, hardware security, and protection against physical tampering."
    b. Action: This involves instructing the AI to assess the physical security measures surrounding industrial control systems, which are crucial for safe operations.

Human Factor Vulnerabilities: Involves human error or behavior that compromises security. This can include weak password practices, falling for phishing scams, or insider threats.

1. Phishing Vulnerability Assessment:
    a. Prompt: "Analyze recent email and communication logs for signs of phishing attempts, such as suspicious sender addresses, misleading links, or requests for sensitive information."
    b. Action: This prompt directs the AI to look for indicators of phishing, a common attack vector exploiting human error.
2. Password Security Analysis:
    a. Prompt: "Evaluate the strength and complexity of user passwords across the network, identifying weak passwords that could be easily guessed or cracked."
    b. Action: This prompt guides the AI to assess the robustness of password policies and identify potentially vulnerable accounts due to weak password practices.

Cryptography Vulnerabilities: Weaknesses in encryption algorithms or their implementation, which can lead to compromised data confidentiality and integrity.

1. Encryption Algorithm Strength Analysis:
    a. Prompt: "Review the encryption algorithms used in this software for any outdated or weak cryptographic methods, such as MD5 or DES, and identify potential vulnerabilities."
    b. Action: This prompt directs the AI to focus on identifying the use of outdated or weak encryption algorithms known to be vulnerable.
2. Cryptographic Key Management Assessment:
    a. Prompt: "Evaluate the cryptographic key management processes in this system, checking for issues like hard-coded keys, insufficient key lengths, or insecure key storage practices."
    b. Action: The AI is tasked with analyzing how cryptographic keys are managed, looking for common vulnerabilities in key generation, storage, and destruction.

Environmental Vulnerabilities: Relates to external threats that are not under direct control, like natural disasters or large-scale power outages, which can impact cybersecurity.

1. Climate Impact Analysis for Infrastructure:
    a. Prompt: "Evaluate this region's infrastructure resilience against environmental factors such as extreme weather conditions, rising sea levels, or increased frequency of natural disasters due to climate change."
    b. Action: This prompt directs the AI to assess how well infrastructure can withstand evolving environmental challenges, focusing on climate impacts.
2. Disaster Preparedness and Response Evaluation:
    a. Prompt: "Assess the effectiveness of this area's disaster preparedness and response plans for environmental emergencies, such as earthquakes, floods, or wildfires."
    b. Action: This prompt instructs the AI to evaluate emergency preparedness and response plans for various environmental emergencies.

V. Conclusion

This paper aimed at providing an in-depth review of prompt engineering within the cybersecurity domain. It is founded on a comprehensive analysis of existing literature, drawing from a diverse array of sources to present a detailed overview of the current state and advancements in AI. The purpose was to provide a holistic understanding of AI models, including LLMs, and how generative AI, through prompt engineering, can be utilized for cyber-defense.

The LLMs and the wide usage of prompt engineering will soon become an integral part of security assessments. Security experts can use prompt engineering to collect and assess the data and get insights about the security threats in advance. Prompt engineering will be embedded into the security practice to scan, analyze and understand code, and detect weak points, potential malware, and security breaches. Once the analysis and triaging are complete, the security professionals can formulate and evaluate a plan to remediate the vulnerability by applying software updates, firewalls, and encryption methods. Furthermore, prompt engineering will help in summarizing the findings, creating presentations to convey the observations and remediation steps.

LLMs offer the advantages of rapid processing, uniformity, and ongoing learning. Nevertheless, they are not foolproof. There are significant drawbacks, including the possibility of false positives, missing complex vulnerabilities, and the reliance on the availability of quality training data.

Cybersecurity is a dynamic and intricate field, and it is vital to integrate LLM approaches with other security strategies. This includes using traditional security tools, continuous network monitoring, and leveraging human expertise. This comprehensive approach forms the backbone of an effective cyber-defense strategy. Moreover, it's imperative to maintain ethical considerations in the application of LLMs within cybersecurity to prevent any unintentional outcomes or biases that could influence decision-making processes.

Numerous studies are currently being conducted to harness the power of AI in proactively identifying and remedying security flaws within systems. These efforts aim to leverage AI's advanced capabilities to enhance cybersecurity measures and improve overall system resilience. This paper, along with the various studies it references, serves as a foundational resource for future research in this field. It lays the groundwork for further exploration and development, aimed at advancing the application of AI in cybersecurity.

References

[1] S. Sakaoglu, “KARTAL: Web Application Vulnerability Hunting Using Large Language Models: Novel method for detecting logical vulnerabilities in web applications with finetuned Large Language Models”.

[2] J. Wei et al., “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.” arXiv, Jan. 10, 2023. Accessed: Jan. 07, 2024. [Online]. Available: http://arxiv.org/abs/2201.11903.

[3] S. Moskal, S. Laney, E. Hemberg, and U.-M. O’Reilly, “LLMs Killed the Script Kiddie: How Agents Supported by Large Language Models Change the Landscape of Network Threat Testing.” arXiv, Oct. 10, 2023. Accessed: Jan. 07, 2024. [Online]. Available: http://arxiv.org/abs/2310.06936.