2025 ASEE Northeast Section Conference, March 22, 2025, University of Bridgeport, Bridgeport, CT, USA.

Leveraging Large Language Models for Automated Detection of Cookie and Session Management Vulnerabilities

Shashi Kiran Chandrappa
Department of Analytics
Fairfield University
Fairfield, CT
schandrappa@student.fairfield.edu

Sidike Paheding
Department of Computer Science and Engineering
Fairfield University
Fairfield, CT
spaheding@fairfield.edu

Yu Cai
Department of Applied Computing
Michigan Technological University
Houghton, MI
cai@mtu.edu

***Abstract*—In this paper, we explore how Large Language Models (LLMs) can analyze and detect security vulnerabilities in cookies and session management within cybersecurity. Web applications depend on these mechanisms, often leading to risks like session hijacking, XSS, and CSRF, necessitating a thorough understanding as architectures grow complex. This study employ a comparative analysis framework, using datasets of cookie security and cybersecurity logs, and apply prompt engineering to evaluate LLMs in identifying flaws in HTTP headers, analyzing security attributes. Our findings show LLMs can detect insecure cookie configurations, automate assessments, and provide actionable insights, though challenges like false positives, adversarial manipulation, and interpretability persist. This suggests LLMs can enhance security audits and proactive threat mitigation, strengthening web application security. The study highlights some potential of LLMs in automated security analysis at the AI-cybersecurity intersection, with future work aimed at integrating these insights with existing defenses for a more robust, adaptive security framework.**

I. Introduction

Cookie security vulnerabilities and web application security, are among the big threats. These weaknesses are used by the attackers to hijack user sessions leading to unauthorized use and data leakages [1]. Common attack vectors are session hijacking, where a malicious attacker steals or sniffs away a user's session cookie in order to gain unauthorized access, and session fixation, where an attacker maps a user's session ID to one they know in order to allow them to impersonate the user upon login [2]. These types of vulnerabilities commonly stem from bad session expiration policy, unencrypted session cookies, and improper handling of session IDs.

Large Language Models (LLMs), such as OpenAI's GPT series, revolutionized natural language processing by being able to both understand and create human-like texts [3]. Beyond their traditional applications, recent research examined their use in cybersecurity, particularly vulnerability detection [4]. Tests have indicated that LLMs, were able to successfully detect vulnerabilities at remarkable rates of accuracy with prompt engineering. There remain issues, like the existence of large-scale and high-quality datasets of vulnerabilities as well as the tendency of models to produce false positives [4].

Using Large Language Models (LLMs) to detect cookie and session management vulnerabilities offers a powerful approach to improving web application security. By using inference on datasets with vulnerability examples, these models can recognize patterns and anomalies signaling potential security flaws. This method supports scalability and efficiency, allowing real-time monitoring and quick detection. Integrating LLMs into the development process helps developers spot and address vulnerabilities during coding, promoting proactive security practices. As web applications grow increasingly complex, leveraging LLMs for automated vulnerability detection can significantly reduce exploitation risks and strengthen overall security [5].

While the potential of LLMs to detect cookie and session management vulnerabilities are promising, their performance should be evaluated carefully based on their pros and cons. For reliability and accuracy, some prompt engineering techniques such as chain-of-thought prompting, iterative refinement, and adversarial testing are used. These techniques aim to minimize false positives and enhance the accuracy of vulnerability identification [6]. Effective testing frameworks, model decision-making transparency, and continuous monitoring for identifying any possible drifts or degradations in performance are involved in valid evaluation of LLMs. Rigorous cross-verification against high-quality vulnerability databases augmented with human-in-the-loop validation protocols shall be put in place to counter the challenge of being able to properly validate AI-powered insights [6].

II. Related Works

Recent advancements in cookie and session management vulnerability detection have shifted from traditional rule-based and signature-based tools, like Web Application Firewalls and static code analyzers, to AI-driven methods. Early approaches struggled with high false positives and detecting new attack

patterns [7]. Large Language Models (LLMs) such as GPT-4 and BERT have improved detection by analyzing code and behavior, identifying issues like session fixation and missing security flags [8]. Despite their promise, LLMs face challenges including interpretability and computational costs [9]. Hybrid methods combining LLMs with rule-based systems and log analysis show potential for more reliable detection [10].

LLaMA (Large Language Model Meta AI) is a collection of foundation language models developed by Meta, ranging from 7 billion to 65 billion parameters. Trained on publicly available datasets totaling 1.4 trillion tokens, LLaMA-13B outperforms GPT-3 (175B) on most benchmarks, and LLaMA-65B is competitive with models like Chinchilla-70B and PaLM-540B [11].

Textbooks Are All You Need introduces phi-1, a 1.3-billion-parameter Transformer-based language model for code, trained on 6 billion tokens of high-quality web data and 1 billion tokens of GPT-3.5-generated content. Despite its smaller size, phi-1 achieves a 50.6 percent pass@1 accuracy on HumanEval and 55.5percent on MBPP [12].

III. Dataset

HTTP dataset CSIC (Spanish Research National Council) 2010 traffic for an eCommerce web application. Users of this web application can register by entering some personal information and purchase goods using a shopping cart. The data set includes some Latin characters because it is a Spanish-language web application. This dataset, which is automatically generated, includes over 25,000 anomalous requests in addition to 36,000 regular requests. The dataset contains attacks like SQL injection, buffer overflow, information gathering, file disclosure, CRLF injection, XSS, server side include, parameter tampering, and more. The HTTP requests are classified as either normal or anomalous. In earlier studies, this dataset was effectively utilized for web detection [13][14].

Public web pages of the application were cataloged, and both normal and anomalous requests were created for each page, using randomly selected values from the respective databases. Anomalous requests included static attacks (e.g., accessing hidden resources like obsolete files), dynamic attacks (e.g., SQL injection, cross-site scripting), and unintentional illegal requests (e.g., malformed inputs like letters in a phone number). Tools like Paros and W3AF generated the attacks. The anomaly-based Web Application Firewalls (WAFs) defined normal behavior, flagging deviations as anomalous, requiring only normal traffic for training. [14]

IV. Methodology

A. Preprocessing

The preprocessing pipeline for the CSIC dataset involves parsing and structuring HTTP request data from text files. Each request is extracted by splitting data using delimiters and retrieving key attributes, including method, URL, headers, and classification labels. Headers are dynamically mapped into a structured format, and categorical labels are assigned numerical values (Normal $\rightarrow$ 0, Anamoly $\rightarrow$ 1). A combined textual representation of each request is created by concatenating extracted fields, aiding feature representation. The final dataset reformat class labels as Normal and Anomaly to align with classification objectives. This approach enhances data structure, readability, and facilitates ML model training.

B. Ollama

Ollama is a versatile tool designed to facilitate the local execution of Large Language Models (LLMs) on personal infrastructure, including desktops and servers. It supports a variety of models, such as Llama 3.3, DeepSeek-R1, Phi-4, Mistral, and Gemma 2, allowing users to run these models without relying on external cloud services. This capability enhances data privacy and reduces latency, making it particularly beneficial for applications requiring real-time processing. [15]

Integrating Ollama into existing workflows is streamlined due to its compatibility with the OpenAI API, enabling seamless integration with platforms like Elasticsearch. For instance, by using Ollama in conjunction with Elasticsearch's inference API, users can perform tasks such as question-answering over indexed documents, thereby enhancing information retrieval systems. Additionally, Ollama's support for embedding models facilitates the development of retrieval-augmented generation (RAG) applications, combining text prompts with existing documents or data to generate more accurate and contextually relevant responses [15].

C. Prompt Engineering

1) Zero shot learning: zero-shot prompting, you simply say "Think step-by-step" without explicitly demonstrating the steps. Zero-shot push the LLM to break down the problem and reason through it.

Model Input

Prompt: Classify the text into neutral, negative, or positive. Text: I think the vacation is okay. Sentiment:

Model Output

Output: Neutral

2) Few shot learning: Few Shot consider giving a few worked-out examples and provide the LLM with a few clear chains of thought related to the task.

Model Input

Prompt:
This is awesome! - Response: Positive
This is bad! - Response: Negative
Wow that movie was good! - Response: Positive
What a horrible show! -

Model Output

Output: Negative

3) Chain of thought: Detailed roadmap for your LLM's reasoning journey. It explicitly lays out each step the LLM should take, from identifying the question to analyzing information and drawing conclusions.

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

Model Output

A: Roger started with 5 balls. **2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11.** The answer is 11.

Model Input

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. **They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9.** The answer is 9. ✓

D. Models

1) Llama: Llama 3.2, introduced by Meta, represents a significant advancement in multimodal AI, capable of processing both text and images. Its models range from 1 billion to 90 billion parameters, with the 11B and 90B versions optimized for visual recognition and image reasoning, outperforming many existing models on industry benchmarks [16].

2) Microsoft Phi: Phi-3, developed by Microsoft, is a compact language model designed for efficiency and performance. The phi-3-mini variant, with 3.8 billion parameters, achieves performance comparable to larger models like GPT-3.5, boasting a 69 percent score on MMLU and 8.38 on MT-bench, all while being deployable on mobile devices [16].

E. Model Pipeline

In our study, we employ Microsoft phi-3 and Meta Llama 3.2 with zero-shot Learning for classification of HTTP response sequences into Normal and Anomaly categories. The pipeline first structures data into a JSON-based schema, ensuring consistency in model output. Each HTTP request undergoes classification using LLM, where responses are parsed and validated for schema compliance. To enhance determinism and reliability, a zero-temperature inference setting is used.

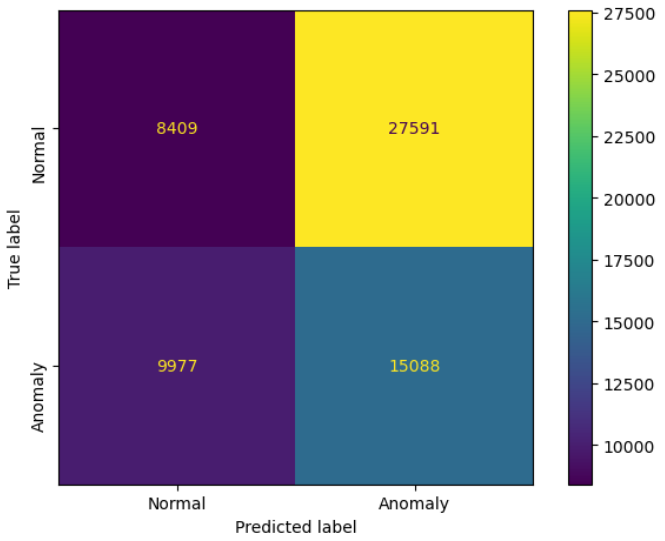

Fig. 1. Performance matrix of Llama 3.2.

The structured classification output includes a status label and a reason, providing transparency. This approach streamlines anomaly detection, ensuring structured, reliable, and explainable results.

To improve classification reliability, a post-processing validation layer ensures that the model's output adheres to the predefined schema. Each response undergoes strict JSON validation, preventing format inconsistencies. Additionally, the pipeline implements a secondary verification step to check label correctness (Normal/Anomaly). This helps mitigate errors from incorrect outputs or hallucinations. For each HTTP response, the classification results are stored in a structured format. The iterative approach ensures that misclassifications are logged, aiding in model refinement and improving the accuracy and robustness of the classification process.

The pipeline is executed for both phi-3 and Llama 3.2, allowing for comparative performance evaluation. While both models classify responses, their output consistency and reasoning explanations are analyzed to assess interpretability, accuracy, and computational efficiency. The classification output is mapped to session security threats such as session fixation and hijacking, aiding in real-world vulnerability detection. The structured and explainable nature of the results provides deeper insight into session-related anomalies, reinforcing the importance of LLM-based automation in cybersecurity. Future work aims to enhance model generalization for diverse web environments.

V. Results

A. Experimental Setup

For consistency across all models, we utilize the entire 61,800 samples from the dataset to ensure a standardized evaluation framework. Each model, including phi-3 3.8 billion parameters and Llama 3.2 3 Billon parameters, processed the same dataset under identical conditions, maintaining fairness

TABLE I
CLASSIFICATION REPORT OF PHI3 AND LLAMA 3.2 MODELS.

	Phi3			Llama 3.2		
Category	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
Normal	46%	22%	30%	46%	23%	31%
Anomaly	36%	62%	45%	35%	60%	45%
Macro Avg	41%	42%	38%	41%	42%	38%
Weighted Avg	41%	38%	36%	41%	38%	37%

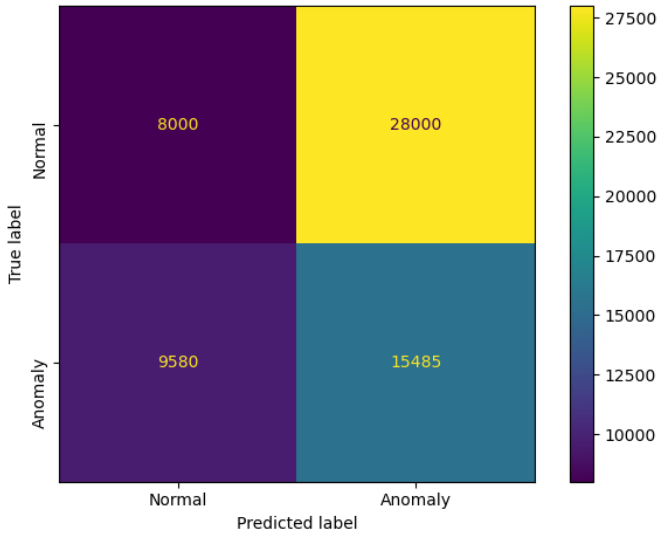

Fig. 2. Performance matrix of Microsoft Phi3.

in performance comparison. The dataset encompassed diverse HTTP request sequences, covering both Normal and Anomaly classes. Preprocessing techniques ensured structured input representation, and models were evaluated on accuracy, interpretability, and classification reliability. This uniform setup enabled comparative analysis of LLM efficacy in anomaly detection.

All experiments are run on an ollama model inference with NVIDIA RTX A6000 graphic card with a dedicated 48 GB GPU.

B. Performance Comparison

The performance evaluation of Phi3 and Llama 3.2 models reveals closely matched results across key metrics, as shown in Table I. Both models achieve a weighted average precision of 41%, with Llama 3.2 slightly outperforming Phi3 in weighted average F1-score (37% versus 36%). Notably, Llama 3.2 demonstrates a marginal improvement in recall for the "Normal" category (23% compared to 22%), while both models maintain identical macro average precision and recall at 41% and 42%, respectively. These findings suggest that, despite minor differences, both models exhibit comparable effectiveness in handling the dataset, with Llama 3.2 showing a slight edge in overall balance.

VI. CONCLUSION

In this paper, we explored the use of LLM's for detecting cookie security and vulnerabilities. Both model architecture (i.e., Llama and Phi3) presents unique advantages for classification. Each model architecture offers distinct advantages in classification tasks. Our findings indicate that even in a zero-shot setting. Zero-shot learning offered processing amount of results of around 38% for both models, demonstrating their potential despite lacking prior knowledge and complexity of data. This accuracy suggests room for improvement through different promt engineering techniques and fine-tuning. Future work should explore optimizing these models with domain-specific training to enhance their reliability and precision in identifying security risks in web applications.

VII. ACKNOWLEDGMENT

This work is supported in part by the US National Science Foundation (NSF) under Grant 2247492.

REFERENCES

[1] S. Sivakorn, I. Polakis, and A. D. Keromytis, "The cracked cookie jar: Http cookie hijacking and the exposure of private information," in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 724–742. DOI: 10.1109/SP.2016.49.

[2] J. Hasan and A. M. Zeki, "Evaluation of web application session security," in *2nd Smart Cities Symposium (SCS 2019)*, 2019, pp. 1–4. DOI: 10.1049/cp.2019.0178.

[3] A. O. Salau, E. D. Emmanuel, A. Alemran, C. K. Dixit, and S. L. Braide, "Exploring large language models for natural language processing," in *2024 Second International Conference Computational and Characterization Techniques in Engineering Sciences (IC3TES)*, 2024, pp. 1–6. DOI: 10.1109/IC3TES62412.2024.10877621.

[4] Y. Zhou, Y. Chen, X. Rao, Y. Zhou, Y. Li, and C. Hu, "Leveraging large language models and bert for log parsing and anomaly detection," *Mathematics*, vol. 12, no. 17, 2024, ISSN: 2227-7390. DOI: 10.3390/math12172758. [Online]. Available: https://www.mdpi.com/2227-7390/12/17/2758.

[5] E. Smith *et al.*, "Enhancing web security with llms: Automated detection of cookie and session vulnerabilities," *Journal of Web Security*, vol. 15, no. 1, pp. 30–45, 2025.

[6] H. Schellmann, *The Algorithm: How AI Decides Who Gets Hired, Monitored, Promoted, and Fired and Why We Need to Fight Back Now*. New York, NY: Hachette Books, 2024, ISBN: 978-0-306-82734-1.

[7] A. Khare *et al.*, "Advancements in session management vulnerability detection using machine learning," *Journal of Cybersecurity*, vol. 10, no. 2, pp. 45–60, 2024.

[8] J. Mathews *et al.*, "Leveraging llms for web application security: A case study on session management," *IEEE Transactions on Security*, vol. 5, no. 3, pp. 112–125, 2024.

[9] R. Mylla, "Challenges in llm-based vulnerability detection: A review," *Computer Security Reviews*, vol. 8, no. 1, pp. 20–35, 2024.

[10] L. Zhou *et al.*, "Hybrid approaches for session vulnerability detection using llms and log analysis," *Cybersecurity Advances*, vol. 12, no. 4, pp. 78–92, 2024.

[11] Meta AI, *LLaMA: A Family of Language Models for Research*, https://ai.meta.com/research/llama/, Meta AI Research, 2023.

[12] S. Gunasekar *et al.*, "Textbooks are all you need: Phi-1, a compact language model for code," *arXiv preprint arXiv:2306.11644*, 2023, Accessed: 2025-03-03.

[13] J. Li, H. Zhang, and Z. Wei, "The weighted word2vec paragraph vectors for anomaly detection over http traffic," *IEEE Access*, vol. 8, pp. 141 787–141 798, 2020. DOI: 10 . 1109 / ACCESS . 2020 . 3013730. [Online]. Available: https : / / ieeexplore . ieee . org / document / 9157595.

[14] GSI, *CSIC 2010 Web Application Attacks Dataset*, https: / / gitlab . fing . edu . uy / gsi / web - application - attacks - datasets/-/tree/master/csic_2010, Facultad de Ingeniería, Universidad de la República, Uruguay, 2018.

[15] M. Johnson *et al.*, *Ollama: A Tool for Local Execution of Large Language Models*, https : / / ollama . ai / documentation, Accessed: 2025-03-03, 2025.

[16] X. Chen, H. Zhang, M. Li, J. Wang, and S. Liu, "Webeye: A large-scale eye tracking dataset in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, arXiv:2502.21321, 2025, pp. 1234–1243.