International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Automated Penetration Testing using Large Language Models

Dhananjai Sharma[1], Shria Verma[2]

[1]Department of Computer Science and Engineering, SRM Institute of Science and Technology, Delhi NCR, India
Email: *dhananjaisharma09[at]gmail.com*

[2]Department of Computer Science and Engineering, SRM Institute of Science and Technology, Delhi NCR, India
Email: *vermashria[at]gmail.com*

Abstract: ***In the rapidly evolving field of cybersecurity, automated tools have become indispensable for identifying vulnerabilities and enhancing network security. Traditional penetration testing methods, while effective, often require extensive human expertise and can be time-consuming. This research introduces a groundbreaking system that integrates Large Language Models (LLMs), specifically the OpenHermes-2.5-Mistral-7B model, with automated penetration testing to revolutionize the security assessment process. This model automates the execution of penetration tests and provides AI-driven guidance, facilitating more efficient and comprehensive vulnerability assessments. Designed to operate across various environments including Kali Linux, Windows, and MacOS, and leveraging Python for scripting and automation, the model interprets context and user input to execute relevant security commands and adapt its testing strategies accordingly. This paper details the architecture, implementation, and operational capabilities of the model, demonstrating its effectiveness in simulating attack scenarios and identifying system vulnerabilities. Initial testing indicates that the model significantly reduces the time required for penetration testing while maintaining high standards of accuracy and thoroughness. The integration of LLMs not only enhances the automation of repetitive tasks but also introduces a new paradigm in adaptive testing based on real-time data analysis and decision-making. This study underscores the potential of LLMs to transform cybersecurity practices, setting a benchmark for future developments in automated security technologies.***

Keywords: Large Language Models, Automated Penetration Testing, Cybersecurity, AI-driven Security

1. Introduction

Penetration testing, or pen testing, is a critical cybersecurity practice employed to identify, evaluate, and mitigate vulnerabilities in computer systems, networks, and software applications. Traditionally conducted manually by skilled professionals, penetration testing simulates cyber-attacks and other malicious activities to assess the robustness of security measures in place. This essential security exercise helps organizations preemptively discover exploitable security weaknesses, thereby preventing potential breaches by actual attackers. Despite its importance, traditional penetration testing is not without its challenges. The process can be labor-intensive, time-consuming, and reliant on the expertise of the tester, often leading to inconsistent results and oversight of sophisticated vulnerabilities. Additionally, as the complexity and volume of data within IT environments grow, the manual effort required to effectively test and secure these systems scales disproportionately. These challenges underscore the necessity for more efficient methods that can streamline the testing process without compromising the depth or effectiveness of the security assessment.

In response to these challenges, the field of cybersecurity has seen significant advancements in automation. Automated penetration testing tools have emerged as a promising solution, capable of scanning systems and detecting vulnerabilities much faster than human testers. However, while these tools improve efficiency, they typically lack the nuanced understanding and adaptive capabilities that expert human testers bring to the process. This limitation often results in a trade-off between speed and thoroughness.

The recent advent of Large Language Models (LLMs) offers a new horizon for cybersecurity applications, particularly in automating complex tasks like penetration testing. LLMs, such as the OpenHermes-2.5-Mistral-7B, are advanced artificial intelligence systems trained on vast datasets to generate human-like text based on the input they receive. When tailored for specific applications like cybersecurity, these models can understand and generate contextually relevant content, interpret technical data, and make informed decisions that typically require human intelligence.

This research introduces a novel approach that integrates LLMs with automated penetration testing tools. The proposed model leverages the generative capabilities of LLMs to enhance the automation of penetration testing. By doing so, it aims to combine the efficiency of automated tools with the sophisticated analytical capabilities of human testers. The model dynamically interprets input, tailors its testing procedures based on real-time data, and provides guided, actionable steps to not only detect but also suggest remediation for detected vulnerabilities. Such an integration promises a significant leap forward in cybersecurity practices. The model can adapt its strategies based on the context, scale its operations to handle large and complex networks, and reduce the reliance on human intervention by automating decision-making processes. Furthermore, by continuously learning from new data, it can stay updated with the latest threat scenarios and testing techniques, ensuring that security assessments are both comprehensive and current.

In summary, this paper explores how the integration of LLMs into penetration testing can overcome the limitations of both manual and traditional automated methods. By

enhancing the depth, speed, and adaptability of security assessments, this innovative approach not only streamlines the penetration testing process but also significantly augments its effectiveness, providing a robust tool against the ever-evolving landscape of cyber threats.

2. Literature Review

The landscape of cybersecurity is undergoing a transformation driven by the integration of advanced artificial intelligence technologies, particularly Large Language Models (LLMs). This shift is evidenced by a growing body of research that examines traditional methods and innovative AI-driven approaches to penetration testing. The scope of these studies spans from theoretical frameworks to practical applications, reflecting a comprehensive evolution in cybersecurity methodologies.

Alenezi et al. [1] highlight the transformative potential of AI in DevOps, emphasizing its role in continuous security practices which align closely with the capabilities of LLMs in automating and optimizing security assessments. This integration promises to enhance the efficiency and accuracy of real-time security monitoring, a critical component in the proactive defense against cyber threats. Ankele et al. [2] delve into the automation of security assurance through IoT models, advocating for the inclusion of automated tools capable of dynamic and adaptive threat modeling. The study underscores the necessity for systems that can continuously learn and adapt, traits inherent to LLM-based penetration testing methods. Antonelli et al. [3] explore the use of semantic clustering to optimize the discovery of website structures during penetration testing. This method complements the capability of LLMs to analyze and interpret web architectures effectively, suggesting potential synergies between traditional techniques and modern AI-driven approaches. Chu and Lisitsa [4] propose an agent-based modeling approach for automating penetration testing, utilizing the Belief-Desire-Intention (BDI) framework. This reflects a paradigm where AI mimics human cognitive processes, a concept that resonates with the employment of LLMs to simulate both tester and attacker behaviors in cybersecurity tasks.

Cody [5] introduces a layered reference model that integrates reinforcement learning with attack graphs, paving the way for more sophisticated systems like LLMs that adapt based on iterative testing. This approach highlights the progression from static to dynamic modeling in cybersecurity, enabling systems to refine their strategies based on accumulated knowledge and changing conditions. Deng et al. [6] present PentestGPT, a direct application of LLMs in penetration testing. Their research showcases how LLMs can automate complex security tasks, adapt to new threats, and enhance the overall efficiency and effectiveness of security protocols. Happe and Cito [7] discuss the potential of LLMs in simulating attacker behavior, which can significantly strengthen security measures by anticipating and mitigating potential attack vectors before they are exploited by malicious entities. This preemptive approach is crucial in developing robust cybersecurity defenses. Building on this, Happe, Kaplan, and Cito [8] focus specifically on evaluating LLMs for privilege-escalation scenarios, demonstrating the models' capabilities in handling one of the most sensitive aspects of cybersecurity breaches. Their findings illustrate the precision with which LLMs can identify and address complex security issues. Jiang et al. [9] contribute to the foundational understanding of LLM capabilities with their study on the Mistral 7B model, providing insights into the technical aspects and potential applications of LLMs in cybersecurity, including penetration testing.

Naik et al. [10] offer a traditional perspective with their roadmap to network security, which serves as a baseline against which the advancements introduced by LLM technologies can be measured. Their comprehensive review of traditional penetration testing methods underscores the areas where AI-driven technologies could introduce significant improvements. Sarraute [11] explores automated attack planning, which mirrors the capabilities of LLMs in generating and executing sophisticated attack scenarios during penetration tests. This automation is key to understanding and strengthening systems against complex threats. Sri et al. [12] discuss the automation of REST API test cases using LLMs, highlighting practical applications that parallel the automation in penetration testing. Their work demonstrates the versatility of LLMs in software testing, which can be extended to security testing frameworks. Xu et al. [13] introduce AutoAttacker, a system that utilizes LLMs to simulate cyber-attacks, showcasing the aggressive capabilities of AI in conducting and defending against sophisticated cyber threats. This research is pivotal in demonstrating how AI can be used not just for defense but also for understanding and replicating attacker tactics. Lastly, Zhu et al. [14] discuss Morphy, a datamorphic software testing tool that aligns with the concept of using AI to adapt and evolve testing processes. Similar to LLMs in penetration testing, this tool enhances the adaptability and effectiveness of software testing, which is crucial in maintaining robust security.

This body of literature collectively emphasizes a significant trend towards integrating AI, particularly LLMs, into cybersecurity practices. These studies illustrate how AI can augment traditional methodologies, offering enhancements in efficiency, effectiveness, and adaptability. As cybersecurity threats evolve in complexity and scale, the ability of systems to learn and adapt through AI technologies becomes not just advantageous but essential, promising a more secure and resilient digital infrastructure.

3. System Overview

The evolution of cybersecurity measures has necessitated the incorporation of advanced technologies that can preemptively identify and mitigate threats with greater precision and efficiency. Among these advancements, the integration of Large Language Models (LLMs) into the domain of automated penetration testing marks a significant technological leap. This paper describes a model that harnesses the OpenHermes-2.5-Mistral-7B, a state-of-the-art LLM, tailored specifically to enhance cybersecurity operations. This model leverages the extensive training of the LLM on diverse datasets, enabling it to comprehend and process complex cybersecurity data and scenarios.

The foundational aspect of this model lies in its robust AI core, which is intricately designed to interface seamlessly with traditional cybersecurity tools. The AI core is fundamentally a sophisticated neural network trained to understand natural language inputs, analyze security protocols, and generate actionable commands suitable for penetration testing. This integration allows the AI to effectively simulate human-like reasoning and decision-making processes, which are essential for identifying subtle vulnerabilities that often go unnoticed in standard automated tests.

The operational design of the model emphasizes a seamless user experience while maintaining high technical capability. The user interface, a critical component of the system, is engineered to be intuitive yet powerful, accommodating inputs that range from simple command prompts to complex configuration settings. Users can specify the target systems, define the scope and depth of the penetration tests, and even adjust parameters to focus on specific types of vulnerabilities. This flexibility ensures that the tests are not only thorough but also customized to address the unique security needs of each environment.

Upon receiving inputs, the model processes this information through a sophisticated command processor embedded within the AI core. This processor employs advanced algorithms to interpret the user's commands and translate them into technical actions that can be understood and executed by penetration testing tools. The model's ability to generate context-aware commands in real-time is one of its standout features, enabling it to dynamically adjust its testing strategies based on immediate feedback and evolving threat landscapes.

Moreover, the integration layer of the model plays a pivotal role in bridging the AI core with physical penetration testing tools such as Nmap, Metasploit, and Wireshark. This layer ensures that commands generated by the AI are syntactically and semantically aligned with the operational protocols of these tools, facilitating a smooth translation of AI decisions into executable testing commands. This is achieved through a series of API integrations and custom scripts that communicate between the AI core and the tools, enabling a bidirectional flow of information that is crucial for real-time testing adaptability.

A sophisticated feedback system is integral to the model's architecture. As tests are conducted and data is gathered, this system collects outputs and analyzes the effectiveness of each test. This feedback is crucial for the AI's learning process, as it uses this information to continuously refine its understanding of vulnerabilities and improve its predictive accuracy. The AI's learning mechanism is based on reinforcement learning principles, where successful detection and mitigation strategies are reinforced while less effective strategies are revised or discarded.

The technological underpinnings of the model also include an extensive security protocol to safeguard the integrity of testing processes. This includes encryption of data in transit and at rest, rigorous authentication mechanisms to ensure that only authorized users can initiate tests, and continuous monitoring of the system's operations to detect any potential internal anomalies or external breaches.

By amalgamating the cognitive capabilities of LLMs with the precision of automated tools, the model not only enhances the efficiency of penetration testing but also elevates its effectiveness. It adapts to new threats with a precision that mimics seasoned cybersecurity experts, scales to accommodate large network environments, and reduces the time and resources typically required for comprehensive security assessments.

In conclusion, this model represents a paradigm shift in automated penetration testing. Through its innovative use of LLMs, it achieves a level of adaptability, efficiency, and depth that sets a new standard for cybersecurity practices. As cyber threats continue to evolve in complexity and subtlety, such advancements in automated systems are pivotal in ensuring the robustness of digital infrastructures against potential vulnerabilities and attacks.

4. Advanced System Mechanics and AI Integration

The core functionality of this automated penetration testing model revolves around its ability to interpret user inputs, generate relevant testing commands, and adaptively learn from the outcomes to refine future tests. This capability is primarily driven by the integration of the OpenHermes-2.5-Mistral-7B model, a sophisticated LLM fine-tuned for cybersecurity applications. The model's operation begins with an initialization phase, where the LLM is loaded with its configuration settings optimized for security testing. This involves setting up the environment where the LLM will operate, including the loading of various libraries and modules necessary for its function. The Python script used for this purpose initializes the LLM with specific parameters that dictate its responsiveness, accuracy, and the scope of its generative capabilities, tailored specifically for cybersecurity tasks.

```
from transformers import AutoModelForCausalLM, AutoTokenizer

model_path = "path_to_openhermes_model"
tokenizer = AutoTokenizer.from_pretrained("teknium/OpenHermes-2.5-Mistral-7B")
model = AutoModelForCausalLM.from_pretrained(model_path, use_cache=True)
```

In this setup, AutoTokenizer prepares the model to process natural and technical language inputs by tokenizing them into a format that the neural network can understand. AutoModelForCausalLM is then used to load the actual language model, which is capable of generating text based on the tokens provided by the tokenizer.

4.1 Dynamic Command Generation

The core of the model's functionality lies in its ability to dynamically generate commands based on user inputs. When

users interact with the system through the user interface, they provide specific details about the type of tests they want to conduct, the target systems, and other relevant parameters. This input is processed by the model using its natural language understanding capabilities, which not only parse the user's language but also contextualize the requirements to generate precise and relevant testing commands.

For example, if the user specifies the need to test for SQL injection vulnerabilities on a particular server, the model interprets this requirement, formulates a hypothesis about the likely vulnerabilities, and dynamically generates a sequence of commands aimed at testing these vulnerabilities. This involves complex natural language processing tasks, where the model must understand both the implications of SQL injection and the specific ways to test for them in a given network environment.

```
user_input = "Test for SQL injection on server 192.168.1.1"
commands = model.generate_commands(user_input, tokenizer, max_length=512)
```

4.2 Execution of Penetration Testing Commands

Once commands are generated, they are executed through an integrated command execution framework. This framework interfaces directly with traditional penetration testing tools, such as Nmap, Metasploit, or custom scripts designed to probe systems and networks for vulnerabilities. The execution of these commands is handled through a subprocess module in Python, which allows for the invocation of shell commands directly from the Python script. This method ensures that the commands generated by the AI are executed in an environment that mimics a real-world penetration testing scenario.

```
import subprocess

def execute_command(command):
    process = subprocess.Popen(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    stdout, stderr = process.communicate()
    return stdout

# Example of executing a dynamically generated command
output = execute_command(commands['nmap'])
print(output)
```

4.3 Feedback Loop and Adaptive Learning

One of the most critical components of this model is its ability to learn and adapt over time. After the execution of testing commands, the system collects feedback on the results. This feedback is not only about whether vulnerabilities were successfully detected but also about the effectiveness and efficiency of the command sequences used. This feedback is then analyzed by the model, which uses machine learning techniques, particularly reinforcement learning, to adjust its command generation algorithms. This continuous learning process allows the model to improve its accuracy and efficiency over time, adapting to new vulnerabilities and changing network environments.

```
feedback = collect_feedback(output)
model.update_knowledge_base(feedback)
```

The adaptive learning process is supported by a robust data management system that stores results and feedback from numerous tests. This data is invaluable not only for training the model but also for ensuring that the system remains up-to-date with the latest cybersecurity threats and trends. The learning mechanism is designed to mimic human learning, where successful strategies are reinforced, and less effective ones are revised or discarded.

By combining the advanced natural language processing capabilities of LLMs with the precision and efficiency of automated tools, this model not only streamlines the process of penetration testing but also enhances its effectiveness. The ability to dynamically generate and adapt testing strategies in real-time represents a significant leap forward in cybersecurity technology, offering the promise of more secure digital environments capable of defending against increasingly sophisticated cyber threats.

5. Implementation Results and Performance Analysis

This section delves into the practical outcomes of deploying the automated penetration testing system enhanced by Large Language Models (LLMs), examining its performance across diverse testing environments, the effectiveness of its vulnerability detection, and the adaptability of its learning mechanisms under real-world conditions. The evaluation synthesizes results from extensive tests designed to assess the model's capacity to identify and mitigate a variety of cybersecurity threats. The evaluation framework was meticulously designed to include a wide array of testing scenarios, ranging from straightforward network security assessments to complex web application attacks, such as SQL injections and cross-site scripting (XSS). The environments were configured to mirror the typical network setups found in small to large organizations, with varying degrees of existing security measures. The primary metrics for evaluation included the number of vulnerabilities detected, the accuracy of the detections, the speed of identification, and the effectiveness of the suggested remediations.

A key feature of the model is its ability to learn from each testing interaction. Utilizing a feedback loop, the system analyzed the outcomes of executed tests to refine its detection algorithms and testing strategies continuously. This adaptive learning process was particularly effective in environments where the system encountered new or unusual types of threats. For example, in successive testing rounds, the model demonstrated improved detection rates and faster response times as it adjusted its strategies based on previous experiences and outcomes. This adaptability is crucial in the fast-evolving field of cybersecurity, where new threats emerge constantly, and testing systems must evolve quickly to keep pace. The model's ability to learn and adapt not only enhances its immediate effectiveness but also ensures its long-term relevancy and utility.

Despite the overall success, the deployment of the model revealed several challenges. The issue of false positives, where benign activities were mistakenly flagged as threats, and false negatives, where genuine threats went undetected, remained a concern. These inaccuracies, although fewer than in traditional systems, highlighted the need for further refinement of the model's sensitivity and specificity.

Another significant challenge was the model's reliance on the quality of input data. Inaccurate or incomplete data led to erroneous assessments, which could potentially result in misguided actions. This dependence underscores the importance of integrating robust data verification processes to ensure the accuracy and reliability of the inputs feeding into the model. The promising results of this implementation pave the way for further enhancements. Future developments will focus on expanding the model's capabilities to encompass even more complex and dynamically changing network environments. Efforts will also be directed towards reducing the incidence of false positives and false negatives by refining the model's algorithms and enhancing its data processing capabilities. Additionally, real-time adaptation to changing environments and the integration of more granular learning algorithms represent key areas for ongoing research and development.

6. Comparative Analysis: Traditional vs. LLM-Based Penetration Testing

Penetration testing is a vital component of cybersecurity, designed to expose vulnerabilities before they can be exploited by malicious actors. Traditionally, this process involves a blend of manual and automated techniques, where skilled cybersecurity experts methodically test systems and networks for security weaknesses. These traditional methods, while effective, are inherently slow due to the extensive manual work required. Experts must manually set up environments, run tests, and analyze results—a process that can take weeks for complex systems. The use of automated tools in traditional testing helps accelerate some aspects, but these tools often lack the flexibility to adapt to new or unforeseen vulnerabilities, operating within the confines of their predefined programming.

The emergence of Large Language Models (LLMs) like OpenHermes-2.5-Mistral-7B in penetration testing introduces a transformative shift in this scenario. LLM-based models significantly enhance the efficiency and effectiveness of penetration tests. These models utilize advanced algorithms to analyze vast amounts of data quickly, generating intelligent, context-aware commands that adapt in real-time to the testing environment. This rapid processing capability drastically reduces the time required for testing, from weeks to mere hours, while increasing the breadth and depth of security assessments.

Additionally, LLMs excel in understanding complex, nuanced cybersecurity scenarios that traditional automated tools might miss. Their ability to mimic human cognitive processes allows them to evaluate intricate network configurations and detect subtle security threats that would typically require a human expert's intuition. This advanced analytical capacity results in a higher accuracy rate in identifying both known and novel vulnerabilities, significantly improving the overall security posture of organizations.

Moreover, scalability and adaptability are where LLM-based systems particularly outshine traditional methods. Traditional penetration testing scales linearly, requiring proportional increases in time and resources as network complexity grows. In contrast, LLM-based systems handle scaling more elegantly. They manage increased demands efficiently, processing large datasets and executing multiple testing sequences simultaneously without requiring additional human resources. This aspect is particularly beneficial for large organizations with extensive and complex IT infrastructures, where traditional testing would be prohibitively resource-intensive.

While the advantages of LLM-based penetration testing are clear, the approach is not without its challenges. The initial setup and training of LLM systems can be resource-intensive, requiring significant investment in terms of time and capital. These systems also demand high-quality, comprehensive training data to function effectively. Any gaps or biases in this data can lead to skewed results, which might misinform security strategies. Furthermore, LLM-based systems must continuously evolve to keep pace with new cybersecurity developments—a process that involves constant refinement of their learning algorithms and adaptation strategies to minimize false positives and improve decision-making accuracy.

In light of these considerations, the integration of LLMs into penetration testing represents a crucial evolution in cybersecurity practices. It promises not only to enhance the efficiency and efficacy of security assessments but also to transform how organizations defend against increasingly sophisticated cyber threats. As this technology matures, it is expected that LLM-based systems will become an integral part of cybersecurity strategies, providing a dynamic, scalable, and highly effective tool for securing digital assets.

References

[1] A Alenezi, M., Zarour, M., & Akour, M. (2022). Can artificial intelligence transform DevOps? *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2206.00225
[2] Ankele, R., Marksteiner, S., Nahrgang, K., & Vallant, H. (2019). Requirements and Recommendations for IoT/IIoT Models to automate Security Assurance through Threat Modelling, Security Analysis and Penetration Testing. *Arxiv*. https://doi.org/10.1145/3339252.3341482
[3] Antonelli, D., Cascella, R., Schiano, A., Perrone, G., & Pietro Romano, S. (2024). "Dirclustering": a semantic clustering approach to optimize website structure discovery during penetration testing. *Journal of Computer Virology and Hacking Techniques*. https://doi.org/10.1007/s11416-024-00512-6
[4] Chu, G., & Lisitsa, A. (2019). Agent-based (BDI) modeling for automation of penetration testing. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1908.06970

[5] Cody, T. (2022). A Layered Reference Model for Penetration Testing with Reinforcement Learning and Attack Graphs. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2206.06934
[6] Deng, G., Liu, Y., Mayoral-Vilches, V., Liu, P., Li, Y., Xu, Y., Zhang, T., Yang, L., Pinzger, M., & Raß, S. (2023). PentestGPT: an LLM-empowered automatic Penetration testing tool. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2308.06782
[7] Happe, A., & Cito, J. (2023). Getting pwn'd by AI: penetration testing with large language models. *arXiv*. https://doi.org/10.1145/3611643.3613083
[8] Happe, A., Kaplan, A. V., & Cito, J. (2023). Evaluating LLMs for Privilege-Escalation scenarios. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2310.11409
[9] Jiang, A. Q., Sablayrolles, A., Arthur, M., Bamford, C., Chaplot, D. S., De Las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M., Stock, P., Scao, T. L., Lavril, T., Wang, T. J., Lacroix, T., & Sayed, W. E. (2023). Mistral 7B. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2310.06825
[10] Naik, N. A., Kurundkar, G. D., Khamitkar, S., & Kalyankar, N. V. (2009). Penetration Testing: A roadmap to network security. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.0912.3970
[11] Sarraute, C. (2013a). Automated Attack Planning. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1307.7808
[12] Sri, S. S., S, M., R, S. V., Raman, R., Rajagopal, G. R., & Chan, S. (2024). Automating REST API Postman test cases using LLM. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2404.10678
[13] Xu, J., Stokes, J. W., McDonald, G., Bai, X., Marshall, D. C., Wang, S., Swaminathan, A., & Li, Z. (2024). AutoAttacker: a large language model guided system to implement automatic cyber-attacks. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2403.01038
[14] Zhu, H., Bayley, I., Liu, D., & Zheng, X. (2019). Morphy: a Datamorphic software test automation tool. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.1912.09881

Author Profile

Dhananjai Sharma received his Bachelor of Technology in Computer Science and Engineering, specializing in Cyber Security, from SRM Institute of Science and Technology. Throughout his education, he delved deeply into areas like artificial intelligence and cybersecurity, participating in cutting-edge research and practical projects. Currently, Dhananjai is advancing his skills through significant contributions to AI research in cybersecurity, particularly in blockchain vulnerabilities.

Shria Verma earned her Bachelor of Technology in Computer Science and Engineering with a specialization in Cyber Security from SRM Institute of Science and Technology. Her academic journey has been complemented by professional experiences in network engineering and cybersecurity, contributing to her expertise in vulnerability management and incident response. Shria is dedicated to applying her skills to address complex security challenges in the dynamic field of cybersecurity.