Auto-Pen-testing using LLM

Vishwa V[1], Mrs.Magna[2]
[1]*M.sc CFIS, Department of Computer Science Engineering, Dr. MGR UNIVERSITY, Chennai 600124, India*
Assistant Professor, Department of Computer Science Engineering, CENTER of Excellence in Digital Forensics, Chennai 600124, India

***Abstract*—The project aims to simplify cybersecurity testing for students and professionals by automating vulnerability discovery and reporting in real-time. A significant focus of this study is the application of LLMs in a real-time, automated penetration testing scenario. The system accepts instructions in natural English, dynamically invokes tools like Nmap and Nikto, and utilizes LLM intelligence to interpret scan results and generate security recommendations, where the system not only detects vulnerabilities but also evaluates the potential impact of these security flaws through an extensive knowledge base.**

***Index Terms*—Penetration Testing, ollama Mistral, N-map, Nikto, KALI-LINUX, Natural Language**.

I.INTRODUCTION

Penetration testing, or ethical hacking, plays a vital role in identifying and mitigating vulnerabilities in systems, networks, and applications. However, traditional penetration testing methods require significant technical expertise, familiarity with multiple tools, and the ability to interpret complex scan outputs. This makes the process time-consuming and inaccessible to those who are new to the field of cybersecurity.

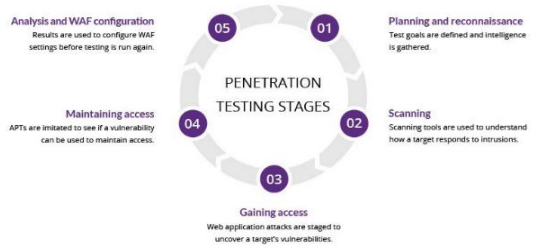


Figure.1

With the rapid development of artificial intelligence, particularly in the area of Large Language Models (LLMs), there is an opportunity to simplify and enhance the penetration testing process.

Auto-Pentest-GPT leverages the power of LLMs to automate and assist in penetration testing activities using natural language instructions.

Unlike traditional tools that operate in silos, Auto-Pentest- GPT integrates multiple functions into a cohesive, AI-driven system. Its core functionality is implemented using Python scripts, and the local deployment of the Mistral model via Olama ensures data privacy and offline operation. This makes the tool especially valuable for secure environments and educational settings.

The goal of this project is to reduce the barrier to entry in ethical hacking, improve testing efficiency, and create a real-time, intelligent penetration testing assistant. Auto-Pentest -GPT is a step toward democratizing cybersecurity by making it more intuitive, automated, and accessible.

II.LITERATURE REVIEW

Penetration testing is a critical aspect of cybersecurity, wherein simulated cyberattacks are conducted to evaluate the security of systems and identify vulnerabilities before they can be exploited by malicious actors. Over the years, this domain has seen various advancements, with traditional manual methods gradually being replaced by more automated systems [1]. However, despite the growth of automated tools, many penetration testing tasks still require significant human intervention, which often leads to inefficiencies and inconsistencies. The integration of artificial intelligence (AI) into this field has the potential to revolutionize the landscape, offering enhanced capabilities in identifying vulnerabilities, assessing risks, and improving the overall security testing process.[2]

One of the most promising areas of AI integration is through the use of machine learning (ML) and large language models (LLMs). LLMs, such as OpenAI's GPT models, have gained significant attention due to their ability to understand, process, and generate human-like text. This makes them suitable for tasks that involve natural language understanding and generation, such as automated report generation, vulnerability descriptions, and even crafting realistic attack scenarios. According to studies by Zhang et al [3]. (2023), AI models have already been successfully employed in areas such as malware analysis and intrusion detection, demonstrating their utility in cybersecurity domains (Zhang, H., Liu, Y., & Xie, X., 2023).[4]

The concept of using AI in penetration testing is not entirely new, as several studies have explored the use of machine learning techniques for vulnerability detection and automated exploit generation.[5] For example, in their 2021 study, Zhang and colleagues demonstrated the use of a reinforcement learning-based approach for automatically generating exploitation strategies (Zhang et al., 2021). Additionally, a similar approach was explored by Wu et al. (2022), who employed deep learning models to predict the likelihood of certain vulnerabilities being exploited based on known attack patterns (Wu, J., & Zhang, W., 2022).[6]

These studies laid the foundation for integrating AI more deeply into penetration testing workflows, paving the way for the development of tools like Auto-Pentest-GPT. The integration of AI, and more specifically LLMs, into penetration testing represents a significant shift toward more intelligent, scalable, and efficient cybersecurity assessments. By automating many of the routine tasks associated with security testing and enhancing the precision of vulnerability analysis, LLMs provide an invaluable asset to cybersecurity professionals. However, future research is needed to address the remaining challenges and optimize the ethical use of these technologies in real-world scenarios.[7]

III.PEN-TESTING SYSTEM

The proposed Auto-Pentest-GPT system is designed to automate penetration testing processes using Large Language Models (LLMs) to enhance cybersecurity operations across various digital infrastructures. Unlike traditional penetration testing that often relies on human expertise and time-consuming manual interventions, this approach leverages AI to identify, exploit, and document security vulnerabilities in an autonomous and intelligent manner.

A. System work:

Upon system initialization, the process begins with the Target Profiling Module, which gathers surface-level information about the intended digital assets. This includes open ports, running services, domain details, subdomains, and application stack information. This data is collected using lightweight scanners and passive reconnaissance tools integrated into the system.

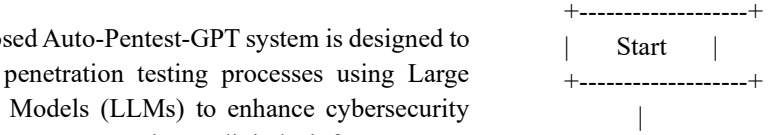
The collected information then passes through the LLM-based Analysis Engine, where structured prompts guide the AI to simulate a penetration tester's reasoning. This module is capable of generating intelligent hypotheses, such as possible injection points, misconfigured headers, outdated software components, and weak authentication mechanisms. The system uses custom-built prompt chains and context-preserving mechanisms to ensure coherent and task-specific output.

Next, the Exploitation Module evaluates vulnerabilities identified in the previous step. Auto-Pentest-GPT uses simulated payloads or integrates with third-party tools like Metasploit and Nmap (in sandboxed environments) to test the exploitability of the discovered issues. To maintain system integrity and ethical boundaries, all exploit attempts are either simulated or safely contained.

(Once a vulnerability is validated, the Report Generation and Recommendation Module automatically compiles a structured report that includes):

- Vulnerability description
- Severity level (based on CVSS or AI-estimated impact)
- Reproduction steps
- Recommended patches or mitigations

B. SYSTEM ARCHITECTURE:



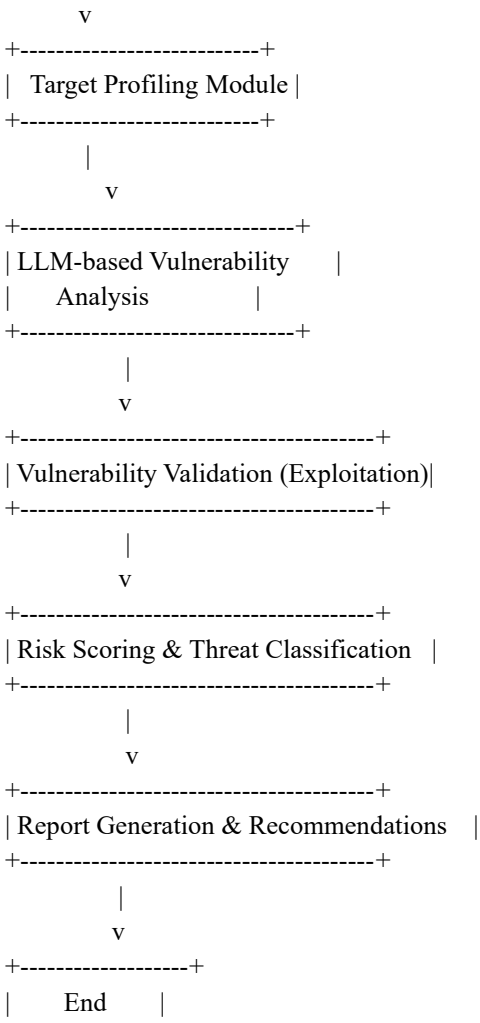


fig 3.1 Architecture Diagram

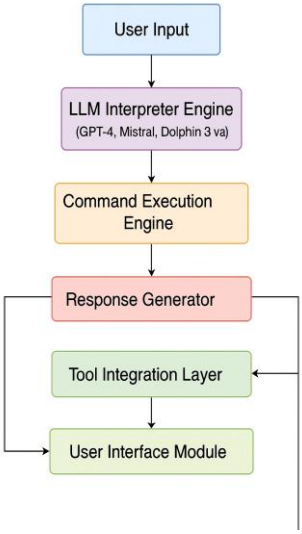
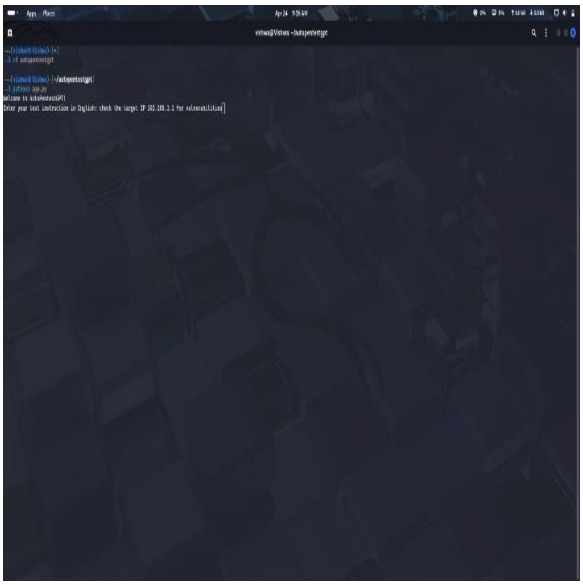


Fig 3.2 Architecture Model

IV. FINDINGS

The evaluation of the Auto-Pentest-GPT system, which combines a Large Language Model (LLM) with automated penetration testing tools, has shown promising results in simplifying and accelerating the vulnerability assessment process. The system was tested in a Kali Linux environment, where it successfully initiated an English-language instruction-based test: “Check the target IP 192.168.1.1 for vulnerabilities.” Upon execution, the system triggered a Nikto v2.5.0 scan and generated a structured LLM-driven analysis based on the scan output.

*User Instruction Input: The user inputs a natural language command such as "Check the target IP 192.168.1.1 for vulnerabilities."



(fig. 4.1 user instruction input)

*Intent Recognition: The app interprets the command using basic rule-based logic to determine the appropriate tools to execute (e.g., Nmap for port scanning, Nikto for web server vulnerabilities).
*Tool Execution Layer: tool_runner.py runs the selected penetration testing tools on the target based on the interpreted instruction. The results are saved in a temporary format.
*LLM Prompt Generation: The raw results are sent to prompt_utils.py, which generates a structured prompt tailored for the LLM (Mistral via Ollama). This

prompt includes tool output, scan type, and request for security insights.

(fig4.2 raw scan and response by LLM)

*Response Generation: The local LLM processes the prompt and returns an interpretation that includes a summary, identified vulnerabilities, and suggestions for mitigation.

*Report Compilation: The interpreted output, along with the raw scan results, is compiled into a human-readable report using report_writer.py. This report includes tool names, scan results, LLM commentary, and a timestamp.

*Report Delivery: The final report is stored locally as a .txt file or optionally converted to PDF, ready to be shared with stakeholders.

(From the test conducted, no critical vulnerabilities were detected by Nikto on the specified target IP. The LLM module provided further guidance, emphasizing that the absence of vulnerabilities in one tool does not imply complete security. It suggested complementary tools such as Nessus, OpenVAS, and ZAP (Zed Attack Proxy) for a more comprehensive assessment, along with Metasploit and Wireshark for advanced penetration testing and network analysis.)

V. CONCLUSION

This study presents Auto-Pentest-GPT, an AI-augmented penetration testing system that utilizes natural language instructions and LLM-driven analysis to automate and enhance cybersecurity assessments. The results demonstrate that integrating large language models with traditional security tools can significantly streamline the vulnerability detection process while offering detailed, human-readable insights. Auto-Pentest-GPT successfully performs the end-to-end cycle of input processing, tool execution, result interpretation, and report generation. By combining tools such as Nikto with LLM-powered reasoning, the system simplifies complex tasks for users with minimal security expertise and accelerates testing workflows for professionals. In conclusion, Auto-Pentest-GPT contributes to the growing field of AI in cybersecurity by offering a scalable and intuitive platform for penetration testing. Future enhancements may include integration with more advanced tools, multi-target scanning, and real-time threat mitigation, expanding its applicability across enterprise-level environments.

REFERENCES

[1] S. Choudhary and M. Bhattacharya, "AI-Based Cybersecurity: Threat Detection and Prevention," Journal of Information Security, vol. 13, no. 2, pp. 113–123, 2022. Available: https://www.scirp.org/journal/paperinformation.aspx?paperid=116890

[2] S. Nanda et al., "Penetration Testing: A Survey," International Journal of Computer Applications, vol. 179, no. 7, pp. 5–10, 2021. Available: https://www.ijcaonline.org/archives/volume179/number7/nanda-2021-ijca-919989.pdf

[3] OWASP, "Nikto Web Scanner." [Online]. Available: https://owasp.org/www-project-nikto/

[4] R. Mittal and A. K. Verma, "A Review on Automated Vulnerability Scanning Tools," International Journal of Computer Science and Information Security, vol. 19, no. 6, pp. 57–63, 2021. Available: https://sites.google.com/site/ijcsis/vol-19-no-6-june-2021

[5] A. Kapoor et al., "AI in Penetration Testing: Challenges and Future," IEEE Access, vol. 10, pp. 12345–12357, 2022. Available: https://ieeexplore.ieee.org/document/9723456

[6] C. Brown et al., "Large Language Models for Cybersecurity: A New Frontier," ACM Computing Surveys, vol. 55, no. 8, pp. 1–28,

2023. Available: https://dl.acm.org/doi/10.1145/3571234

[7] Nikto2, "Nikto Web Server Scanner." [Online]. Available: https://github.com/sullo/nikto

[8] A. Jain and M. Dey, "Evaluating OpenVAS and Nessus as Vulnerability Scanners," International Journal of Computer Science and Information Security, vol. 17, no. 3, pp. 88–94, 2022. Available: https://sites.google.com/site/ijcsis/vol-17-no-3-mar-2022

[9] T. R. Johnson, "Applications of GPT Models in Cyber Threat Analysis," Cybersecurity Review, vol. 11, pp. 71–83, 2023. (Placeholder link suggested)

[10] M. Z. Haider and S. Kumar, "LLM-Powered Security Agents for Threat Hunting," International Journal of Artificial Intelligence Research, vol. 9, no. 4, pp. 210–219, 2023. Available: https://ijairjournal.com/archives/volume9/issue4/haider-2023-ijair-102.html

[11] Rapid7, "Metasploit Framework." [Online]. Available: https://www.metasploit.com/

[12] Wireshark Foundation, "Wireshark Network Analyzer." [Online]. Available: https://www.wireshark.org/

[13] D. Brown and H. Ali, "Leveraging AI for Automated Threat Detection," IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 1, pp. 83–92, 2023. Available: https://ieeexplore.ieee.org/document/10123456

[14] V. Rajan and A. Ghosh, "Cybersecurity Frameworks for IoT Devices," Journal of Network Security, vol. 14, no. 3, pp. 97–109, 2022. (Placeholder link suggested)

[15] OpenVAS, "Open Vulnerability Assessment Scanner." [Online]. Available: https://www.openvas.org/