Introducing PenTest++: an AI-augmented, Automated, Ethical Hacking System

Haitham S. Al-Sinani(ORCID: 0009-0005-0453-3335)
Department of Cybersecurity and Quality Assurance
Diwan of Royal Court, Muscat, Oman

Chris J. Mitchell(ORCID: 0000-0002-6118-0055)
Department of Information Security
Royal Holloway, University of London, Egham, UK.

***Abstract*—Traditional ethical hacking depends on skilled professionals and time-intensive command management, limiting its scalability and efficiency. To help address these challenges, we propose `PenTest++`, an AI-augmented system that combines automation with generative AI (GenAI) to optimise ethical hacking workflows. In this paper, we describe how `PenTest++` operates, present a proof-of-concept prototype, and demonstrate its ability to streamline tasks such as reconnaissance, scanning, exploitation, and documentation. We also provide a balanced discussion of the system's benefits and limitations, including its modular, adaptable design and the importance of ethical safeguards. This research highlights the potential of AI-driven systems to complement human expertise in cybersecurity while emphasising the need for ongoing refinement.**

***Index Terms*—AI, Ethical Hacking, GenAI, ChatGPT, LLM.**

I. Introduction

Ethical hacking [1] plays a critical role in modern cybersecurity, providing organisations with proactive measures to identify and address vulnerabilities before they can be exploited. However, the practice remains highly resource-intensive, requiring advanced expertise and substantial time investments across all stages, from reconnaissance to exploitation. As cyber threats evolve, ethical hackers struggle to keep up with a dynamic landscape and complex environments.

A key limitation in traditional ethical hacking lies in its reliance on skilled professionals to manually execute and interpret intricate processes. This dependency restricts scalability and efficiency, making it difficult to meet the growing demands of increasingly interconnected systems. Moreover, ethical hackers face added challenges in recalling or referencing an ever-growing set of commands across diverse environments. Automation could alleviate some burdens, but even automated solutions struggle with processing large volumes of textual output and extracting actionable insights efficiently.

The emergence of GenAI technologies, or LLMs (Large Language Models), offers significant potential to address these limitations. Tools like ChatGPT[1] excel in streamlining complex tasks, extracting critical data from extensive datasets, and generating actionable insights. Integrating GenAI with automation systems allows ethical hackers to minimise time and cognitive effort associated with manual processes, while maintaining high standards of accuracy and relevance. Furthermore, incorporating AI into ethical hacking enhances accessibility, empowers both novice and seasoned professionals, and reduces the cost and time required for security assessments, benefiting organisations of all sizes. However, AI hallucinations highlight the need for a user-centric approach with human oversight to validate and refine GenAI recommendations.

This paper introduces `PenTest++`, a command-line, AI-augmented automation system designed to enhance ethical hacking workflows. The system integrates GenAI capabilities to assist in automating key tasks, such as reconnaissance, scanning, enumeration, and exploitation, while ensuring user control and adaptability. By referencing and generating appropriate commands dynamically, `PenTest++` reduces the reliance on manual input and supports users in efficiently processing and interpreting complex outputs.

The remainder of this document is organised as follows. Section II defines the research questions and highlights the key contributions of this work. Section III outlines how `PenTest++` operates. Section IV presents the laboratory setup, and section V details a prototype implementation. Section VI discusses the potential benefits, risks and study limitations. Section VII reviews related work, and, section VIII summarises our conclusions and outlines plans for future work.

II. Research Questions and Contributions

A. Research Questions

This study seeks to address the following questions:

1) Is it possible to automate the ethical hacking process effectively in a user-centric manner?
2) How can GenAI be integrated into this automation to enhance efficiency and accuracy?
3) How much human intervention is required to ensure the accuracy and ethical compliance of an AI-driven ethical hacking system?

B. Key Contributions

`PenTest++` advances the field of cybersecurity by addressing critical gaps in automation and AI integration within ethical hacking. The contributions of this study include:

- demonstrating the feasibility of automating the ethical hacking process through an AI-augmented tool;
- introducing a mixed-initiative system that balances AI-driven automation with user oversight, fostering trust and adaptability in ethical hacking;

[1]https://openai.com/blog/chatgpt

- providing a proof-of-concept for AI-driven automation in cybersecurity, with the full intention of making it open-source on GitHub; and
- presenting empirical evidence and insights to support future theoretical and practical advancements in AI-augmented cybersecurity tools.

III. PenTest++ Operation

The operation of `PenTest++` follows a systematic methodology aimed at optimising ethical hacking workflows by combining automation, user oversight, and GenAI support. The `PenTest++` operation is outlined below.

1) **Reconnaissance:** `PenTest++` automates network discovery by executing commands to identify live systems within the target environment. Users select the desired target for subsequent phases.
2) **Scanning & Enumeration:** `PenTest++` executes vulnerability scanning, employing tools such as `nmap` and `gobuster` to identify critical open ports, services, and misconfigurations. GenAI interprets scan outputs, correlates findings with known vulnerabilities, and provides recommendations for targeted enumeration.
3) **Exploitation:** `PenTest++` facilitates exploitation by generating tailored payloads to exploit identified vulnerabilities, such as misconfigured services or insecure functionalities. GenAI offers strategic guidance to craft precise attack sequences, while users dynamically adjust tactics in response to real-time inputs and recommendations provided by the system.
4) **Documentation:** `PenTest++` automates report generation, by leveraging GenAI to produce a comprehensive, penetration testing (PenTesting) report, including logs, methodologies, key findings, and actionable recommendations. GenAI refines the structure and clarity of the documentation, ensuring it provides actionable insights for enhancing the security posture of the tested systems.

IV. Laboratory Setup

A. Development Language

We developed `PenTest++` in `Python 3` for its versatility, rich libraries, AI integration, and adaptability, though alternatives like Bash or Go could also have been used.

B. Physical Host and Virtual Environment Configuration

The experiments utilised a MacBook Pro with 16 GB RAM, a 2.8 GHz Quad-Core Intel Core i7 processor, and 1 TB of storage, supporting virtualisation via VirtualBox 7. The virtual setup included:

1) **Kali Linux VM**: The primary attack platform hosting `PenTest++` for PenTesting.
2) **Linux VM 1**: A 64-bit Debian system with 512 MB RAM, serving as a main target.
3) **Linux VM 2**: Another 64-bit Linux system with 512 MB RAM, acting as an additional target.

A NAT configuration enabled seamless communication between the VMs, simulating a realistic network for testing.

C. GenAI Tool

`PenTest++` utilised the paid ChatGPT-4o for its advanced capabilities (being a leader among the LLMs), preferring an online LLM, over a local one, to benefit from automatic updates. Possible alternatives include Gemini and DeepSeek.

D. ChatGPT-PenTest++ Integration

To integrate ChatGPT programmatically with `PenTest++`, we utilised OpenAI's GPT-4 API, which operates on a subscription-based model[2]. Key `PenTest++` log files and prompts were sent as API payloads, with ChatGPT returning actionable insights.

V. Prototype Implementation

We present a prototype, proof-of-concept implementation.

A. Reconnaissance Module

Reconnaissance, the initial stage of ethical hacking, identifies potential network targets. **PenTest++** begins by determining the IP address and subnet mask of the attacker's machine dynamically. Using this information, the subnet range of the target network is calculated. The total number of hosts in the network is derived using the standard formula $2^{(32-\textbf{CIDR})} - 2$, where $CIDR$ (Classless Inter-Domain Routing) represents the number of bits used for the network portion of the address, and the subtraction of 2 accounts for the network and broadcast addresses. This ensures that all potential IP addresses are identified for subsequent scanning. The system prompts the user to confirm the correctness of the detected range. If incorrect, users can manually input the correct range, and the system recalculates the total number of hosts accordingly. This flexibility ensures adaptability to diverse network setups.

`PenTest++` then actively scans the target network using the `nmap` command (`nmap -sn -T4 subnet`) to detect live hosts, where `-sn` specifies a ping scan to identify active devices without port scanning, `-T4` sets an aggressive timing for faster execution, and `subnet` defines the network range to be scanned. The results display the IP addresses of all active hosts, from which the user can select a target. While `nmap` was the primary tool employed, the design of `PenTest++` is modular, allowing for future integration of other scanning tools such as `arp-scan`, `netdiscover`, or `Masscan`.

To ensure transparency, `PenTest++` displays the exact command being executed, fostering trust and clarity for the user. After scanning, users are prompted to select a target host from the list of active IPs. E.g., if the user selects 192.168.1.7 as the target, control shifts seamlessly to the next module.

B. Scanning & Enumeration Module

This module constitutes a critical phase of the PenTesting process, aimed at identifying open ports, running services, and potential vulnerabilities on target systems. The scanning process in **PenTest++** is automated through the use of `nmap`, a widely adopted tool for network exploration and vulnerability assessment. The system programmatically executes `nmap`

[2]https://platform.openai.com/

scans using Python's subprocess library. The default scan uses the command `nmap -p- -A -T4`, which scans all 65,535 TCP ports for thoroughness, performs service detection, OS identification, and traceroute for target insights.

The scan results are processed to extract key information, including the **port**, which represents the port number and associated protocol (e.g., TCP or UDP); the **status**, indicating the state of the port (e.g., open, closed, filtered); the **service**, specifying the service running on the port (e.g., HTTP, SSH); and the **version**, providing version information and additional details, such as software banners.

The extracted data are presented in a structured tabular format using the `PrettyTable` library. This format facilitates the clear communication of findings and supports decision-making during subsequent PenTesting phases.

Users decide whether to continue enumeration based on scan results, allowing ethical hackers to dynamically adjust the workflow and focus resources on relevant targets.

C. Exploitation Module

This module represents a critical phase in the ethical hacking process, focusing on exploiting vulnerabilities identified during scanning to gain initial footholds in the target system. After the scanning phase, the **PenTest++** system processes open ports and associated services identified on the target. Users are presented with a list of available ports, including details such as the port number and the corresponding service (e.g., HTTP, SSH, or NFS). The system supports an iterative workflow, allowing users to select a port and service for detailed analysis and targeted exploitation. For specific services, `PenTest++` employs tailored attack strategies, as follows.

- **NFS (port 2049)**: If NFS is detected, the system checks for accessible shares and exploitable misconfigurations.
- **FTP (port 21)**: `PenTest++` evaluates the FTP service for anonymous login capabilities or weak credentials.
- **HTTP/HTTPS (port 80)**: HTTP-based services are examined for potential vulnerabilities, including directory traversal, misconfigurations, and local file inclusion (LFI). Identified endpoints, such as `/dev/index.php`, are further investigated for exploitability.
- **SSH (port 22)**: For SSH services, `PenTest++` integrates with ChatGPT to analyse logs and deduce valid credentials or private key information. If necessary, brute-force techniques are employed to crack SSH passwords, ensuring multiple avenues of access are considered.
- **HTTP Proxy (port 8080)**: When proxy services are identified, the system evaluates potential misconfigurations that could allow unauthorised access.

Additional services can readily be integrated and automated due to the modular architecture of the `PenTest++` system.

The module emphasises user flexibility by enabling manual input of credentials, customisation of attacks, and iterative analysis. For example, if a private key is password-protected, users are prompted to provide the associated passphrase. The system integrates ChatGPT to assist in vulnerability analysis, enhancing decision-making during exploitation attempts.

`PenTest++` automates common PenTesting commands while maintaining transparency by displaying the executed commands to the user. This ensures a balance between automation and user oversight, fostering trust and adaptability during the ethical hacking process.

Once access is achieved, the system logs all findings and prepares for the subsequent phase, which involves maintaining and elevating access. This systematic approach ensures that all relevant attack vectors are explored, providing a comprehensive evaluation of the target's security posture.

We will next provide two case studies that demonstrate `PenTest++` in action.

Gaining Access to VM 1: 192.168.1.7

The first case study focuses on VM 1 (`192.168.1.7`), illustrating how the system dynamically automates the steps necessary to exploit a target machine while ensuring that the user remains in control at every stage. This user-centric approach balances efficiency and oversight, empowering the user to make informed decisions while benefiting from a high level of automation.

In the case of VM 1, `PenTest++` first performed an `nmap` scan on `192.168.1.7`. The results identified the following services as potential entry points:

- **FTP (Port 21)**: Anonymous login was enabled, allowing access to files such as `note.txt`.
- **HTTP (Port 80)**: The server displayed the default Apache page, hinting at potential vulnerabilities in Apache 2.4.38.
- **SSH (Port 22)**: No immediate vulnerabilities were evident, but potential attacks could involve brute-forcing credentials or exploiting OpenSSH 7.9p1.

At each step of the reconnaissance process, `PenTest++` presented the findings to the user in a clear and actionable manner, enabling them to select the next course of action.

FTP Attack Vector (Port 21): Using the `PenTest++` system, the FTP service on port 21 was exploited dynamically. The system automatically logged in as an `anonymous` user, retrieved the file `note.txt`, and sent it to ChatGPT for an in-depth analysis. ChatGPT returned its findings in JSON format, which `PenTest++` parsed and displayed to the user in a tabular format using `PrettyTable`. This user-friendly presentation allowed the user to review and confirm key insights before proceeding.

The analysis of `note.txt` uncovered a hashed password (given below), a potential SQL injection vulnerability via an `INSERT` statement, and authentication credentials (`StudentRegno: 10201321` and `pincode: 777777`). Guided by ChatGPT's recommendations, `PenTest++` utilised `Hashcat` to crack the MD5 hash, revealing the password `student`, as follows:

```
hashcat -m 0 -a 0 cd7350...3fb0bc8 wordlist
```

This command instructs `hashcat` to hash each entry from the selected wordlist using MD5 and check if it matches

the target hash. If successful, it reveals the original password. This cracked password (student), combined with the StudentRegno, provided a viable credential pair for authentication attempts on other services. The process ensured a human-in-the-loop approach by allowing users to modify, pause, or override automation at each step.

HTTP Attack Vector (Port 80): Focusing on the HTTP service, PenTest++ leveraged gobuster to identify hidden directories on the web server, using this command: (gobuster dir -u http://192.168.1.7 -w wordlist). The command uses gobuster in directory brute-forcing mode (dir) to scan the target URL http://192.168.1.7 with a specified wordlist (-w wordlist), while PenTest++ calculates wordlist entries to help users balance accuracy and scanning time. PenTest++ thus successfully discovered three hidden directories on the site: uploads, academy, and phpmyadmin, which could be accessed by appending the directory names to the site URL, e.g.: http://192.168.1.7/academy/. The academy directory contained a login form for an online course system.

The credentials (10201321 and student) were successfully utilised to authenticate to the web application. Within the user profile page, an image upload functionality was identified. Following guidance from ChatGPT, this feature was exploited to upload a PHP reverse shell, disguised as an image file, to facilitate further access to the target system.

PenTest++ auto-generated a PHP reverse shell file containing the code: <?php exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.1.4/6655 0>&1'"); ?> This file was successfully uploaded through the web interface. Simultaneously, the PenTest++ system sets up a Netcat listener on the attack machine (192.168.1.4) on port 6655, using: nc -nvlp 6655. The command disables DNS resolution (-n), enables verbose mode for detailed connection logs (-v), sets the machine as a listener (-l), and specifies port 6655 (-p 6655) to establish shell access to the target VM, which was indeed successfully accomplished and verified through the listener.

Gaining Access to VM 2: 192.168.1.10

We now detail the systematic steps taken to gain access to the second target VM (192.168.1.10), showcasing PenTest++'s capabilities in automating and guiding the exploitation process. As with the previous case study, the user maintained full control, approving each step in the workflow.

Initial Analysis and NFS Exploitation: The nmap scan output identified several open services on 192.168.1.10, including NFS (port 2049), SSH (port 22), and HTTP (ports 80 and 8080). Among these, the NFS service provided the first viable attack vector. PenTest++ dynamically executed showmount -e 192.168.1.10 to identify shared directories, revealing /srv/nfs as accessible.

The system mounted the shared directory at a local mount point (/tmp/nfs_mount_srv) and listed its contents. A file named save.zip was discovered but found to be password-protected. PenTest++ automatically prepared the zip file for password cracking by converting it into a hash format suitable for John the Ripper (JTR). Using the popular rockyou.txt wordlist, JTR successfully identified the password as java101. The file was then decompressed, revealing two files: an SSH private key (id_rsa) and a text file (todo.txt). The todo.txt file contained task notes, offering potential clues about the system's setup and the user (jp). Simultaneously, the private key (id_rsa) provided a possible means of SSH authentication. However, attempts to use the private key for direct SSH access failed, as additional credentials were required.

HTTP Enumeration on Port 80: Moving to the HTTP service on port 80, PenTest++ leveraged its automated functionalities to conduct directory enumeration using gobuster (gobuster dir -u http://192.168.1.10 -w wordlist). The system autonomously identified accessible directories, including /app, /config, and /database. Upon discovering the /config directory, PenTest++ automatically initiated a download of the config.yml file, dynamically prompting the user for confirmation before proceeding.

Following the download, PenTest++ executed a thorough scan of the file for sensitive content using a predefined list of keywords. During this process, the system detected database credentials, including the password (I_love_java) associated with the bolt user. The results were seamlessly formatted into a user-friendly table using PrettyTable, ensuring clear presentation and efficient decision-making.

Exploiting HTTP Service on Port 8080: When analysing the HTTP service running on port 8080, PenTest++ detected the presence of the BoltWire CMS and identified a Local File Inclusion (LFI) vulnerability. Leveraging guidance from ChatGPT, the system generated an LFI payload targeting the /etc/passwd file: http://192.168.1.10:8080/dev/index.php?p=action.search&action=../../../../../../../etc/passwd.

PenTest++ prompted the user for approval before opening http://192.168.1.10:8080 in the default browser to facilitate manual inspection. The system recommended that the user register an account to establish an authenticated session, ensuring the payload could be executed effectively. Upon execution, the payload successfully retrieved the /etc/passwd file, exposing system usernames such as jeanpaul and root. The process demonstrated the system's blend of automation and user-guided vulnerability exploitation.

SSH Access via Port 22: PenTest++ leveraged information from LFI exploitation to target the SSH service on 192.168.1.10, using the jeanpaul account and a private key (id_rsa) secured with the passphrase I_love_java. To ensure security, it adjusted the private key's permissions with the command chmod 600 id_rsa, where 600 restricts the file to read and write access for the owner only. The system then executed ssh -i id_rsa jeanpaul@192.168.1.10, where -i id_rsa specifies the private key file, jeanpaul is the username, and 192.168.1.10 is the target IP address. This successfully established an SSH session, compromising the target system.

To enhance user experience, PenTest++ launched the SSH session in a new terminal window using supported terminal emulators like gnome-terminal or xterm. By doing so, it ensured the main terminal remained undisturbed, allowing users to interactively proceed with other tasks, e.g. generating pentest reports or conducting further investigations.

If the private key was unavailable or ineffective, PenTest++ employed sshpass to attempt password-based authentication using previously discovered credentials. Users were presented with a list of available usernames, enabling them to choose the most suitable option. The system dynamically adapted to the selected authentication method, ensuring all viable access points were tested.

In cases where authentication attempts failed, PenTest++ integrated the Hydra password-cracking tool to perform brute-force attacks on the SSH service. The system allowed configuration of parameters such as target usernames, thread count, verbosity, and custom password lists, enabling tailored attacks. For example, to brute-force the root account, PenTest++ executed the following Hydra command, allowing users to tailor the inputs as needed: `hydra -l root -P rockyou.txt ssh://192.168.1.10 -t 4 -f -V`. This command methodically tested passwords against the root account, with Hydra reporting any successful login credentials if found. Once valid credentials were discovered, PenTest++ automated the subsequent login process and initiated post-exploitation tasks, such as report generation.

D. Reporting and Documentation

The PenTest++ system automates the creation of a comprehensive PenTesting report, ensuring that ethical hackers deliver detailed documentation for each engagement. Using the findings from the PenTesting activities, PenTest++ interacts with ChatGPT to generate a well-structured and professional report, which can be output in text, JSON, and PDF formats.

PenTest++ prompts the user to confirm whether to generate the report. If approved, it gathers the logs and findings from the penetration test engagement and constructs a detailed prompt for ChatGPT. This prompt includes the necessary metadata, such as the target IP address, the attacker's IP address (Kali machine), and structured log data. The key sections of the report provided by ChatGPT included **Executive Summary**, **Objectives and Scope**, **Methodology** (detailing phases and tools used), **Findings and Vulnerabilities**, **Risk Rating** (categorised into High, Medium, and Low), **Recommendations**, **Conclusions**, and **Appendices** (containing outputs such as Nmap scan results). PenTest++ generates reports in text, JSON, and PDF formats for flexibility.

For VM1 (192.168.1.7), it identified FTP and HTTP vulnerabilities, retrieving hashed credentials via anonymous FTP, cracking the password (student), and using it to authenticate on a web app. An insecure file upload enabled a PHP reverse shell, granting access.

For VM2 (192.168.1.10), vulnerabilities in SSH, NFS, and HTTP were exploited. Misconfigured NFS shares exposed a protected zip file, whose passphrase (I_love_java) was found in a hidden HTTP config. LFI on port 8080 revealed the jeanpaul username, enabling SSH access.

VI. Discussion and Analysis

A. Answers to the Research Questions

We now address the research questions posed.

1) **Is it possible to automate the ethical hacking process effectively in a user-centric manner?** Yes, automation of the ethical hacking process is achievable in a user-centric manner through the integration of systems like PenTest++. By leveraging modular design principles, PenTest++ combines task automation with user oversight, ensuring that ethical hackers remain in control of the process. The system allows users to confirm decisions at key points, adjust strategies, and maintain a clear understanding of the workflow.
2) **How can GenAI be integrated into this automation to enhance efficiency and accuracy?** GenAI can be integrated into automation systems by serving as an analytical assistant that interprets tool outputs, suggests exploitation strategies, and structures reporting. For example, GenAI augments efficiency by parsing large datasets, identifying vulnerabilities, and correlating findings with known exploits. This integration reduces manual effort while maintaining a high degree of precision and adaptability.
3) **How much human intervention is required to ensure the accuracy and ethical compliance of an AI-driven ethical hacking system?** Users must validate AI-generated outputs to prevent errors and ensure ethical alignment with principles like data privacy and responsible disclosure. While PenTest++ automates tasks, it maintains user control with minimal intervention for efficiency and oversight.

B. Benefits and Features

PenTest++ streamlines PenTesting by combining automation with analytical capabilities via GenAI while maintaining user oversight. Key features include:

- **Automated Execution with User Control:** PenTest++ automates scanning, enumeration, and exploitation, reducing manual effort. Users retain control by approving key decisions and dynamically adjusting payloads, authentication methods, or scanning parameters.
- **GenAI-Assisted Analysis:** ChatGPT enhances analytical capabilities by interpreting tool outputs, suggesting exploitation strategies, and identifying vulnerabilities, such as directory traversal or credential leaks.
- **Efficient Data Processing:** Results are structured in tables, enabling quick decision-making. ChatGPT extracts critical insights from large outputs, such as credentials from configuration files.
- **Automated Reporting:** Structured PenTesting reports are generated, including findings and recommendations, in multiple formats (text, JSON, PDF).

- **Modular and Cross-Platform:** The modular design supports integration with additional tools and exploits. Python-based implementation ensures compatibility across Windows, macOS, and Linux.

C. Limitations and Risks

Despite its benefits, PenTest++ has limitations; it lacks full automation, and tasks such as privilege escalation and track-covering fall beyond this study's scope. Ethical concerns arise from automating sensitive tasks, posing misuse risks. The system was tested in a controlled environment, limiting its applicability to diverse systems. Using ChatGPT for analysis raises privacy concerns, requiring legal compliance, while its hallucinations demand human oversight for greater accuracy.

VII. Related Work

The integration of AI in cybersecurity is an active research area, covering intrusion detection and offensive security, including ethical hacking. Despite significant advancements [2]–[8], a fully autonomous PenTesting system remains elusive. **PentestGPT [7]** has been proposed as an LLM-powered PenTesting assistant that leverages Reasoning, Generation, and Parsing Modules for a segmented problem-solving strategy. It follows a human-in-the-loop approach, requiring users to manually enter the target IP, execute suggested commands, and iteratively provide feedback for further guidance. While it aims to address context loss, recent content bias, and hallucinations, its reliance on manual execution greatly limits automation. In contrast, PenTest++ automates predefined PenTesting commands, requiring only user selection and approval. It also begins with network discovery, mimicking real-world PenTesting, whereas PentestGPT requires manual IP input. Ultimately, PenTest++ offers greater automation, whereas PentestGPT serves as a structured ChatGPT wrapper for PenTesting guidance.

Our own earlier research explored GenAI's role in ethical hacking across various phases. In [9], we proposed a conceptual framework for integrating GenAI into PenTesting workflows. Subsequent studies evaluated ChatGPT's effectiveness in controlled Windows [10] and Linux environments [11]. More recently, we examined its application in manual exploitation and privilege escalation [12]. These studies demonstrated GenAI's potential to enhance efficiency, decision-making, and workflow automation from reconnaissance to reporting. Building on this foundation, PenTest++ introduces a user-centric, AI-powered automation system to streamline PenTesting while maintaining human oversight. A more detailed version of PenTest++ is available at [13].

VIII. Conclusions and Further Research

In this paper, we introduced PenTest++, an AI-augmented ethical hacking system designed to streamline and enhance the PenTesting process. We conducted and described a prototype implementation of the system, demonstrating its practical application in automating repetitive tasks, providing real-time analytical support, and maintaining user control throughout ethical hacking workflows. By integrating GenAI, PenTest++ offers penetration testers a robust and adaptive tool to identify and exploit vulnerabilities in target systems.

This study demonstrated PenTest++'s modular design, cross-platform compatibility, and seamless GenAI integration for analysis and guidance. The prototype adapted dynamically to various scenarios while maintaining transparency and user oversight. However, findings stressed the need to balance automation with human expertise to mitigate AI over-reliance and uphold ethical PenTesting principles.

We plan to extend PenTest++ to support privilege escalation, a key post-exploitation task. While this paper focused on qualitative results in controlled settings, further research is needed to evaluate performance in complex environments. Enhancing support for macOS, Android, IoT, wireless security, and mobile threats will improve adaptability. Future work should investigate quantitative metrics (e.g., time saved, AI success rates, user satisfaction) and add comparative analyses with other AI tools. Ethical and operational issues must be addressed, including offline AI use, legal compliance, and mitigation of hallucinations and misuse risks. These efforts will help make PenTest++ practical, secure, and ethical.

References

[1] M. Swanson *et al.*, "Technical guide to information security testing and assessment (NIST SP 800-115)," Special Publication, 2008.

[2] P. Xiong and L. Peyton, "A model-driven penetration test framework for web applications," in *8th International Conference on PST: Privacy, Security and Trust*, 2010, pp. 173–180.

[3] Y. Stefinko, A. Piskozub, and R. Banakh, "Manual and automated penetration testing. benefits and drawbacks. Modern tendency," in *13th International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science*, 2016, pp. 488–491.

[4] F. Abu-Dabaseh and E. Alshammari, "Automated penetration testing: An overview," in *4th international conference on natural language computing, Copenhagen, Denmark*, 2018, pp. 121–129.

[5] A. Happe and J. Cito, "Getting pwn'd by AI: Penetration testing with large language models," in *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2023, pp. 2082–2086.

[6] M. Hassanin and N. Moustafa, "A comprehensive overview of large language models (LLMs) for cyber defences: Opportunities and directions," *arXiv preprint arXiv:2405.14487*, 2024.

[7] G. Deng and et al., "PentestGPT: Evaluating and harnessing large language models for automated penetration testing," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 847–864.

[8] W. Lazarov, P. Seda, Z. Martinasek, and R. Kummel, "Penterep: Comprehensive penetration testing with adaptable interactive checklists," *Computers & Security*, vol. 154, p. 104399, 2025.

[9] H. Al-Sinani, C. Mitchell, N. Sahli, and M. Al-Siyabi, "Unleashing AI in ethical hacking," in *STM '24, Poland, Proceedings*, ser. LNCS, F. Martinelli and R. Rios, Eds. Springer, Dec. 2024, pp. 140–151.

[10] H. Al-Sinani and C. Mitchell, "Unleashing AI in ethical hacking: A preliminary experimental study," Royal Holloway, University of London, Technical Report, Jan 2024, https://pure.royalholloway.ac.uk/files/58692091/TechReport_UnleashingAIinEthicalHacking.pdf.

[11] H. S. Al-Sinani, N. Sahli, C. J. Mitchell, and M. Al-Siyabi, "Advancing ethical hacking with AI: A Linux-based experimental study," in *Proceedings of the Joint National Conference on Cybersecurity (ITASEC & SERICS 2025)*, G. Costa, R. Montanari, M. Carminati, and G. Sciarretta, Eds., vol. 3962. Bologna, Italy: CEUR-WS, February 03–08 2025. [Online]. Available: https://ceur-ws.org/Vol-3962/paper7.pdf

[12] H. Al-Sinani and C. Mitchell, "AI-augmented ethical hacking: A practical examination of manual exploitation and privilege escalation in Linux environments," arXiv, Technical Report, Nov. 2024.

[13] H. S. Al-Sinani and C. J. Mitchell, "PenTest++: Elevating ethical hacking with AI and automation," Feb 2025, arXiv:2502.09484 [cs.CR]. [Online]. Available: https://arxiv.org/abs/2502.09484