AI-Powered Penetration Testing using Shennina: From Simulation to Validation

Stylianos Karagiannis
skaragiannis@ionio.gr
Ionian University
Corfu, Greece and PDM
Lisbon, Portugal

Camilla Fusco
cam.fusco@studenti.unina.it
University of Naples Federico II
Naples, Italy and Montimage
Paris, France

Leonidas Agathos
leonidas.agathos@pdmfc.com
PDM
Lisbon, Portugal and Ionian University
Corfu, Greece

Wissam Mallouli
wissam.mallouli@montimage.com
Montimage
Paris, France

Valentina Casola
valentina.casola@unina.it
University of Naples Federico II
Naples, Italy

Christoforos Ntantogian
dadoyan@ionio.gr
Ionian University
Corfu, Greece

Emmanouil Magkos
emagos@ionio.gr
Ionian University
Corfu, Greece

ABSTRACT

Artificial intelligence has been greatly improved nowadays, providing innovative approaches in cybersecurity both on offensive and defensive tactics. AI can be specifically utilized to automate and conduct penetration testing, a task that is usually time-intensive, involves high-costs, and requires cybersecurity professionals of high expertise. In this research paper, we utilize an AI penetration testing framework to validate, discover and analyze the techniques that were used. To this end, we conducted a validation process in a realistic environment and to collect the relevant datasets from the execution of the cyberattacks. Finally, the behavior of the AI penetration testing was analyzed in order to adapt and upgrade further. Overall, the research paper provides contributions to dataset generation and a methodology to understand the details of the attack simulation.

KEYWORDS

Cybersecurity, Artificial Intelligence, Penetration Testing, Red Teaming, AI-Powered Attacks, Reinforcement Learning

ACM Reference Format:
Stylianos Karagiannis, Camilla Fusco, Leonidas Agathos, Wissam Mallouli, Valentina Casola, Christoforos Ntantogian, and Emmanouil Magkos. 2024. AI-Powered Penetration Testing using Shennina: From Simulation to Validation. In *The 19th International Conference on Availability, Reliability and Security (ARES 2024), July 30–August 02, 2024, Vienna, Austria.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3664476.3670452

ARES 2024, July 30–August 02, 2024, Vienna, Austria

ACM ISBN 979-8-4007-1718-5/24/07
https://doi.org/10.1145/3664476.3670452

1 INTRODUCTION

Artificial Intelligence (AI) has made a great advancement nowadays and more specifically Reinforcement Learning (RL) which has been a leading method. In cybersecurity, RL and AI can play an important role, both for offensive and defensive purposes [4]. Traditionally, penetration testing, is usually performed manually [20]. For example, managing an exploitation database and reporting vulnerabilities requires great effort [3]. Therefore, the need for skilled individuals to perform manual tests is increasing, and it is difficult to find suitable professionals [8]. Therefore, automated penetration testing is becoming increasingly important [2, 5, 20].

Furthermore, new threats arise from Adversarial Machine Learning (AML) in cybersecurity [6, 12]. AML regards the design and execution of adversarial attacks that can disrupt AI systems, leading to incorrect decisions and outcomes. The convergence of AI and cybersecurity has the potential to spark groundbreaking initiatives [15, 17, 18]. Other researchers have also been investigating the usage of AI on offensive procedures [1, 7]. As cybersecurity threats continue to evolve, it is necessary to advance on the methodologies to defend against sophisticated attacks. Traditional penetration testing, crucial for identifying vulnerabilities, often requires extensive manual effort and expertise.

In this research, a testbed for extracting relevant datasets from the cyberattacks is presented and an analysis of the effectiveness of AI-powered attack methods in simulating realistic cyber threats. The research explores the integration of RL. More specifically, Shennina[1]. Shennina, as an automating host exploitation with AI, was adapted to simulate cyberattacks on Metasploitable[2], providing a realistic environment for testing offensive tactics.

The primary objective was to identify and analyze the capabilities of RL, as outlined in the AI4SIM, a component for conducting simulation of advanced and AI-powered attacks. The complete component, namely AI4SIM for simulating AI-powered

[1]https://github.com/mazen160/shennina
[2]https://github.com/rapid7/metasploitable3

attacks, has been proposed from the AI4CYBER project [12] and a complete architecture has been developed to evaluate its functionality. The research focuses on the development of AI4SIM, which revolves around creating an attack simulation solution capable of generating advanced AI-powered attacks. By validating Shennina, this research contributes to AI4SIM by providing results that facilitate its integration into the unified platform. To achieve this, the deployment was equipped with Shennina and distributed agents to mimic real-world attack scenarios and collect datasets. The primary objective of this task is to identify the types of advanced attacks that can be executed using Shennina.

1.1 Contribution

The research paper makes advancements on log, data, and network collection as well as providing an aggregation methodology that is focused on matching the cyberattacks to the MITRE Tactics, Techniques, and Procedures (TTPs). By doing so, the paper addresses the need for researchers to access datasets for further analysis and development of cybersecurity solutions. The key contributions of this research paper are as follows:

(1) The research contributes to the results on the development of the AI4SIM and extracts the benefits for enabling the advanced AI-powered cyberattacks in a realistic environment.
(2) The paper provides information and results on the validation and the effectiveness of the AI4SIM framework by conducting simulations and analyzing the collected datasets. Through this validation, the AI models are to be adapted and further test the ability to create detection rules that will accurately detect and analyze AI-powered attacks. Furthermore, the research paper provides a methodology for validating AI-powered cyberattacks and cyberattack simulations.
(3) Finally, the dataset extraction from the cyberattacks that are being executed can be further exploited.

Overall, the research paper contributes to the advancement of cybersecurity research by providing a practical solution for simulating and analyzing AI-powered cyberattacks. It offers valuable insights into the rule-creation for the detection and mitigation of such threats, thereby enhancing the resilience of critical systems against evolving cyber threats.

1.2 Related work

The main role of AI in cybersecurity has predominantly been focused on the development of new attack methodologies. Yamin et al., 2021 presented a comparative analysis between classical cyberattacks and those powered by AI [22]. They highlight three main types of AI-powered cyberattacks: data misclassification, synthetic data generation, and data analysis.

In another work [16], Nakas et al., developed an AI-powered attack generator that leverages Generative Adversarial Networks (GANs) to fuzz and target the Packet Forwarding Control Protocol (PFCP) in 5G Core networks. Adversarial attacks, leveraging GANs, consist of two networks trained simultaneously for generation and discrimination, have been extensively used in cybersecurity, notably for data generation without explicitly modeling probability density functions [23].

The application of AI in penetration testing, can contribute to the preparation of the defenses of computer networks. Regular penetration testing involves four phases: planning and preparation, detection and penetration, post-exploitation and data exfiltration, and reporting and cleanup [19]. Automated penetration testing, integrating AI techniques like RL, has shown promise. For example, a project explored the applicability of RL in automating penetration testing, using a fast, lightweight, open-source network attack simulator to train and test autonomous agents [11]. The research specifically presents the effectiveness of RL, including Q-learning [21], in finding valid attack paths across different network topologies.

In another research, Happe and Cito, investigated the use of Large Language Models (LLMs) to enhance penetration testing [10]. Their research explores scenarios involving high-level task planning and low-level tasks including vulnerability enumeration and demonstrating the potential usage of AI in penetration testing. Similarly, Ghanem and Chen [9] proposed an AI-based penetration testing system. In addition, Kaloev and Krastev [13] presented that constrained exploration in RL training accelerates learning, improving the performance of the penetration testing. Finally, Maeda and Mimura [14] integrated deep RL on the Empire[3], a post-exploitation framework, to automate post-exploitation activities.

The work distinguishes itself from other approaches by providing an extensive validation of AI-powered penetration testing and the creation of a realistic testbed for extracting attack datasets. Furthermore, MITRE ATT&CK was used as a structured framework to analyze the behavior of the AI-powered cyberattacks and identify the tactics that were executed.

2 METHODOLOGY

This section outlines the validation process for the attack and detection architecture, aimed at validating both the Shennina framework and the architecture itself, which should be capable of detecting attacks attempted on the target.

2.1 Testbed Architecture

The architecture of the testbed is presented in Figure 1. There are two key components which play pivotal roles on the architecture: Shennina, as the AI-powered cyberattack tool, and Metasploitable 3 as a virtual environment intentionally deployed that contain vulnerabilities.

Shennina is being configured to autonomously execute a variety of cyberattacks within the controlled environment. The cyberattacks encompass a range of tactics, including but not limited to buffer overflow exploits, SQL injection, and remote code execution. The dynamic nature of the AI algorithms allows the adaptation of the attack strategies based on the responses received from the target system, making its behavior more sophisticated and challenging.

During the simulation, extensive logging mechanisms are employed to capture detailed information about the attack

[3] https://github.com/EmpireProject/Empire

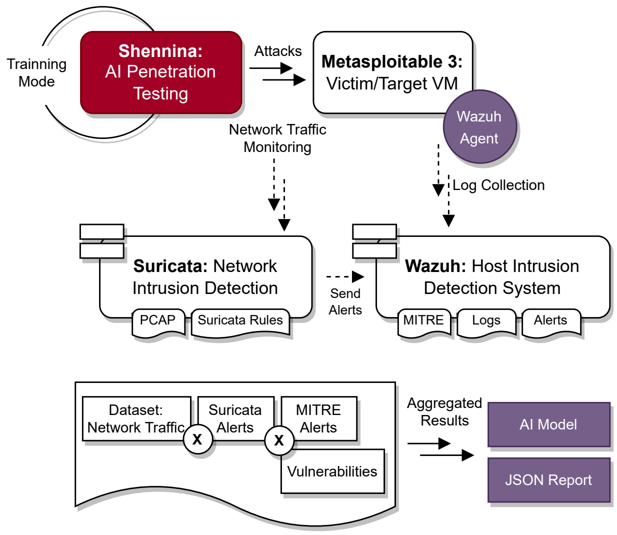

Figure 1: Testbed Architecture and Methodology Flow Diagram.

payloads, commands executed, system responses, and any network traffic generated during the engagements. This comprehensive logging infrastructure serves as the primary source of data for subsequent analysis and validation. Additionally, network traffic monitoring tools are utilized to capture and analyze the data packets exchanged between Shennina and Metasploitable, providing insights into the communication patterns and potential indicators of compromise. Metasploitable is a deliberately vulnerable virtual machine designed with a plethora of service (Table 1) security vulnerabilities, serving as a prime target for exploit testing with Metasploit Framework[4].

The list of running services of Metasploitable are presented in Table 1 providing details on the affected services, corresponding ports, and protocols. For instance, the first row indicates the service GlassFish[5], which operates on ports 4848, 8080, and 8181, utilizing the Hypertext Transfer Protocol (HTTP) protocol.

The collected data as presented in Table 1 can be used to contextualize and categorize the observed attack behaviors. The data are compared against the MITRE ATT&CK framework by matching the observed behaviors with known TTPs documented in the MITRE framework. This process helps in understanding the operation of the cyberattacks which are executed and understand the behavior of the AI cyberattacks regarding the target environment.

The analysis of Shennina and the attack behavior, is processed in this research towards the validation against MITRE TTPs and Suricata signatures. This provides valuable information and results regarding the effectiveness and evasiveness of AI-powered cyberattacks. The process facilitates the refinement and optimization of the approach to develop more robust and sophisticated cyber defense mechanisms. Furthermore, the datasets which are generated during the testing phase serve as valuable resources for training and evaluating AI-based detection mechanisms.

Table 1: Overview of Running Services on Metasploitable 3

Service	Port	Protocol
GlassFish	4848	HTTP
	8080	HTTP
	8181	HTTPS
Apache Struts	8282	HTTP
Tomcat	8282	HTTP
Jenkins	8484	HTTP
IIS - FTP	21	FTP
IIS - HTTP	80	HTTP
psexec	445	SMB
	139	NteBIOS
SSH	22	SSH
WinRM	5985	HTTPS
chinese caidao	80	HTTP
ManageEngine	8020	HTTP
ElasticSearch	9200	HTTP
Apache Axis2	8282	HTTP
WebDAV	8585	HTTP
SNMP	161	UDP
MySQL	3306	TCP
JMX	1617	TCP
Wordpress	8585	HTTP
Remote Desktop	3389	RDP
PHPMyAdmin	8585	HTTP
Ruby on Rails	3000	HTTP

2.2 Shennina

Shennina is an AI-powered penetration testing framework that offers various functionalities, including network and service enumeration, vulnerability assessment, attack path generation, and integration with the Metasploit Framework. Shennina, was developed in Python and was built upon a previous implementation, namely DeepExploit[6].

The AI model from Shennina is trained using a RL approach, which involves interacting with the environment and taking actions based on the current state. The model is trained using a dataset of various cyberattacks, including buffer overflow exploits, SQL injection, and remote code execution. The RL algorithm updates the agent's policy based on the rewards received from the environment, aiming to maximize the cumulative reward over time. The model is evaluated using metrics such as accuracy, precision, and recall, ensuring its effectiveness in detecting and exploiting vulnerabilities.

The simulation initiates its operation by conducting a thorough scan of the target network to pinpoint open ports and active services that might be susceptible to exploitation. Utilizing a pre-existing dataset, Shennina identifies vulnerabilities associated with the discovered ports or services, diligently reporting any findings. In comparison to DeepExploit, Shennina selects reliable remote exploits from the Metasploit Framework, eliminating false positives and ensuring automated remote exploitation. Therefore, the speed is optimized in the training phase, adding

[4]https://github.com/rapid7/metasploit-framework
[5]https://github.com/eclipse-ee4j/glassfish
[6]https://github.com/13o-bbr-bbq/machine_learning_security/tree/master/DeepExploit

post-exploitation capabilities, suggesting potential local root exploits, implementing data exfiltration, and improving exploiting clustering for more relevant exploits. In addition, Shennina includes ransomware simulation, deception detection, and confirmation of target exploitation. As a result, Shennina generates an attack path and generates a file in **.h5* format upon detecting the potential vulnerabilities. The generated attack path optimizes the penetration testing process, leading to efficient access to administrative privileges. The tool then exploits identified vulnerabilities according to the generated path. To conclude the procedure, Shennina generates a detailed exploitation report in Markdown format, documenting key information such as target IP, outcomes, exploit details, and utilized payloads, as an output of the test results.

In exploitation mode, Shennina utilizes gathered data to determine the optimal exploit against the target and initiates post-exploitation actions. Additionally, Shennina offers heuristic mode for automated broad analysis, identifying potential security vulnerabilities based on predefined principles and rules without specific tests for each threat.

2.3 Validation Process of Attack and Detection Architecture

The validation proceeded in the following three phases:

(1) Setup and Training of Shennina. In this initial phase, Shennina was installed and configured. Debugging was performed to address any issues and ensure proper functionality. Additionally, the target machine for the subsequent phases, Metasploitable, was selected due to its widespread use and well-known vulnerabilities. Shennina was then trained using Metasploitable as target.
(2) Testbed Deployment. The second phase involved describing the architecture implemented for evaluating the tool. Two Intrusion Detection Systems (IDS), Suricata[7] and Wazuh[8], were employed to monitor network traffic and verify the occurrence of attacks. This setup aimed to simulate a real attack and defense scenario, providing detailed traffic logs capturing all attempted exploits on the target machine.
(3) Observations and Considerations. In the final phase, the generated files were analyzed. Initially, the focus was on evaluating the effectiveness of the tool simulating the attacks. Subsequently, a detailed examination of the attacks performed was conducted to gain insights into the tactics and techniques employed, identifying the most prevalent ones.

3 VALIDATION RESULTS

The results were extracted and analyzed using a combination of manual testing and automated tools. The criteria for validating reported attacks include accurate fingerprinting of the target system, verification of authentication mechanisms, detection of potential vulnerabilities, development of an attack tree, verification of post-exploitation activities, and generation of a detailed report.

[7]https://github.com/OISF/suricata
[8]https://github.com/wazuh/wazuh

The validation and analysis resulted into the data collected by Suricata to identify relevant traffic, focusing on alerts with the highest severity levels. Among the alerts, one of the most significant findings was the detection of a stack overflow vulnerability, indicating potential exploitation actions. Suricata monitors the packets exchanged between Shennina and Metasploitable throughout the attacks. Whenever a packet exhibits a suspicious pattern, Suricata assigns it a label based on one of its predefined rules. These alerts are aggregated and stored in the eve.json file, which is then transmitted to the Wazuh.

Table 2: Alerts Detected by Suricata (Ports 22 and 23)

Suricata.rule	Suricata.description	MITRE.id
ET SCAN Potential SSH Scan OUTBOUND	Outbound SSH scan detected	Remote System Discovery (T1018)
SURICATA Applayer Mismatch protocol both directions	Protocol mismatch detected in SSH traffic	Data Obfuscation (T1001)
ET SCAN Potential SSH Scan	Potential SSH scan detected	Remote System Discovery (T1018)
ET SCAN Non-Allowed Host Tried to Connect to MySQL Server	Unauthorized host tried to connect to MySQL server	Data from Local System (T1005)
ET SCAN Potential SSH Scan OUTBOUND	Outbound SSH scan detected	Remote System Discovery (T1018)
ET SCAN Suspicious inbound to mySQL port 3306	Suspicious inbound traffic to MySQL port	Data from Local System (T1005)
SURICATA STREAM 3way handshake excessive different SYNs	Network traffic showing excessive different SYN packets during the 3-way handshake process	Network Service Scanning (T1046)

As presented in Table 3 the most frequent network protocols involved IPv4, followed by Address Resolution Protocol (ARP) and IPv6 protocols as well. The distribution of protocols reveals the adversary nature of Shennina in various levels.

Table 3: Table of Network Protocols Distribution During the AI-Powered Cyberattacks

Protocol	Percentage of packets
IPv6	0.2%
IPv4	98.2%
ARP	1.6%
TCP	97.1%
UDP	1.1%

As presented in Table 3 the distribution of network protocols is focusing mostly on specific protocols. The extracted datasets provide the observed attack vectors on the specific testbed setup and experiment analysis. It provides insights into the types of network protocols and their frequency distribution overall.

- IPv4 (Internet Protocol version 4): IPv4 constituted the majority of network traffic, representing 98.2% of the total packets. IPv4 is the fourth version of the Internet Protocol and remains the most widely used protocol for communication over the Internet.

- IPv6 (Internet Protocol version 6): This protocol presented lower usage in the rate of 0.2% of the total packets observed in the network traffic. IPv6 is the most recent version of the Internet Protocol, designed to replace IPv4 and accommodate the growing number of devices connected to the Internet by providing a larger address space.
- ARP (Address Resolution Protocol): ARP accounted for 1.6% of the observed packets. ARP is usually used to map IP addresses to physical Media Access Control(MAC) addresses on a local network.

Within the IPv4 protocol, TCP packets are the majority, comprising the majority of traffic, followed by the UDP (Table 3).

Table 3 provides a breakdown of the distribution of network transport protocols observed in the data. The first column of the table lists the types of network protocols observed in the network traffic data. In this case, there are two protocols listed: Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). The second column of the table indicates the proportion of network packets that correspond to each transport protocol, expressed as a percentage of the total. For example, 97.1% of the packets in the dataset are TCP packets, while UDP packets account for only 1.1% of the total packets.

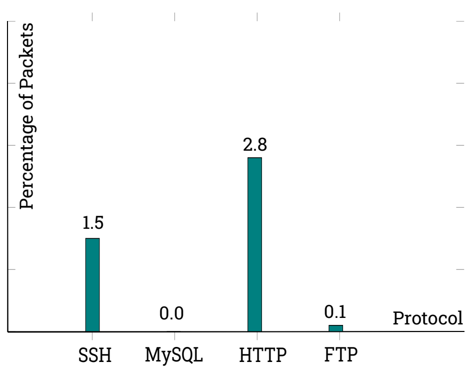

Figure 2: Distribution Percentage of Network Traffic Across Network Protocols

Shennina triggers the calls to the above protocols during the system infiltration. Among these protocols, Secure Shell (SSH) serves as a common target for unauthorized access attempts. Shennina tries to establish SSH connections to Metasploitable, in order to gain remote access and execute commands on the system. This action initiates SSH traffic, which is monitored by Suricata and Wazuh.

Table 2 summarizes alerts from Suricata, focusing on network traffic through port 22, used for SSH. Alerts categorize activities like outbound SSH scans, protocol mismatches, reconnaissance attempts, and unauthorized connections to MySQL (port 3306).

Most of the cyberattacks targeted the HTTP service. By sending HTTP requests to vulnerable web servers, Shennina seeks to exploit weaknesses such as injection flaws, misconfigurations, or authentication bypass vulnerabilities. Similarly, Shennina engage with the File Transfer Protocol (FTP), attempting to transfer files or gain unauthorized access to the system's file system. FTP traffic is monitored for signs of malicious activity, such as unauthorized file transfers or brute-force login attempts.

The investigation extended to analyzing MITRE events and tactics recorded during the testbed period (Table 2 and Table 4). Focus was placed on frequency and distribution of events.

Password guessing and SSH techniques emerged as the most frequent among the captured logs, indicating potential avenues for exploitation. The training phase was divided into two equal time lapses to assess the frequency of attacks. Results indicated that Shennina demonstrated an increase in attack frequency over time, suggesting improvement in exploitation skills during the training phase. Similar observations were made regarding the frequency of Suricata alerts, with approximately 40% of exploits occurring in the first half of the time-lapse. Further analysis highlighted a slight increase in alerts during the second half, indicating a dynamic threat landscape.

Suricata and Wazuh were able to capture the suspicious activities including potential SSH scans, exploitation attempts targeting specific Common Vulnerabilities and Exposures (CVE), including CVE-2016-10174, CVE-2018-19276, CVE-2019-12725, CVE-2022-22947, and unauthorized access attempts to critical services like MySQL. Alerts also cover HTTP protocol violations, shell command execution via HTTP requests, and suspicious patterns in network traffic, such as clear-text passwords in HTTP requests.

The results as presented in Table 4 offers a comprehensive analysis of the tactics which were executed. Each row in the table corresponds to the MITRE tactics, such as Initial Access, Execution, Persistence, etc. Within each tactic, several associated techniques are listed, along with a brief description of each technique's nature. Additionally, the table indicates the effectiveness of Shenina rules in detecting these techniques, along with the frequency of observed instances within the Shenina framework. This structured presentation provides cybersecurity professionals with valuable insights into the capabilities and generates traffic, signatures which can be used to improve detection rules, identify and mitigate potential cyber threats across different stages of an attack lifecycle.

Both Suricata and Wazuh detected the executed cyberattacks, suggesting consistency and supplementary data between the two in the threat detection capabilities. However, questions arose regarding potential undetected attacks by Shennina or overlapping detection by both IDS. Further validation is required to ascertain the accuracy and effectiveness of the tool.

Table 4 demonstrates how the Shenina framework aligns with MITRE's cybersecurity TTPs. Organized by tactic, it outlines specific techniques, their descriptions, corresponding Shenina rules, and their effectiveness. This alignment offers insights into Shenina's ability to detect threats, helping prioritize response efforts and refine detection capabilities. The frequency and type of the observed TTPs, contribute in assessing the attack vectors employed by the AI and the generated datasets can contribute to enhancing the defenses. Towards this direction, for improving the overall TTPs that are employed regular testing and rule refinement based on MITRE TTPs is very important.

Table 4: Distribution of Executed MITRE TTPs from the Macthing of Suricata Alerts to MITRE

*Counter: Number of relevant events triggered by the SIEM and Suricata

MITRE.id	MITRE.tactic	Coverage and MITRE.description	Rule.level	Counter*
	Initial Access	**Covered by T1068, T1068, T1016**		
T1190	Exploit Public-Facing Application	Exploits targeting public-facing applications	12	16
T1068	Exploitation for Privilege Escalation	Various exploits targeting vulnerabilities for privilege escalation	15	7
T1016	Discovery	Detection of general discovery activities	9	1
	Execution	**Covered by T1106, T1203, T1204, T1059**		
T1106	Execution through API	API-based execution attempts	7	9
T1203	Exploitation for Client Execution	Exploits targeting client-side execution vulnerabilities	10	4
T1204	User Execution	User execution activities	1	1
T1059	Command and Scripting Interpreter	Scripting-related security threats	10	2
	Persistence	**Covered by T1074, T1071.001, T1090, T1092, T1094**		
T1074	Data Staged	Suspicious data staging activities on networked systems	7	10
T1071.001	Application Layer Protocol	Protocol anomalies at the application layer	13	1
T1090	Connection Proxy	Network activities involving connection proxies	8	2
T1092	Network Boundary Bridging	Activities bridging network boundaries	4	1
T1094	Protocol Command Decode	Protocol command decoding anomalies	10	1
	Privilege Escalation	**Covered by T1068, T1106, T1020, T1018**		
T1018	Remote System Discovery	Remote system discovery attempts	12	2
	Defense Evasion	**Covered by T1005, T1045**		
T1005	Data from Local System	Sensitive data transmitted over HTTP and suspicious network activities related to database servers	12	11
T1045	Obfuscated Files or Information: Software Packing	Attempts to obfuscate files or information using software packing techniques	5	1
	Credential Access	**Covered by T1074, T1040, T1132, T1043, T1078**		
T1040	Network Sniffing	Network sniffing activities, including potential password exposure	8	7
T1132	Data Encoding	Data encoding activities, including shell command execution attempts	3	3
T1043	Commonly Used Port	Activities related to commonly used ports in security incidents	6	1
T1078	Default Credentials	Default credential usage	8	1
	Discovery	**Covered by T1046, T1069.001, T1016, T1018**		
T1046	Network Service Scanning	Network scanning activities targeting various services	8	6
T1069.001	Operating System Discovery	Network scans aimed at identifying operating systems	4	1
	Lateral Movement	**Covered by T1046, T1105, T1092, T1133**		
T1105	Remote File Copy	Unauthorized file copying activities from remote systems	12	2
T1133	External Remote Services	External remote service connections	9	2
	Collection	**Covered by T1005, T1074, T1213, T1094**		
T1213	Data from Information Repositories	Attempts to gather information from system repositories	6	4
	Command and Control	**Covered by T1041, T1090, T1505, T1046**		
T1041	Exfiltration Over C2C Channel	Attempts to hide or obscure data transmission over the network	10	7
T1505	Web Shell	Web shell activities	1	1
T1046	Service Scanning	Network scanning activities targeting specific services	8	1
	Exfiltration	**Covered by T1041, T1005**		
	Impact	**Covered by T1068, T1001, T1132, T1016**		
T1001	Data Obfuscation	Attempts to hide or obscure data transmission over the network	10	7

Table 4 provides in details the rule coverage for MITRE tactics and techniques, with counts indicating technique frequency. Validation revealed reconnaissance instances, such as detecting web server errors from the same source IP, emphasizing the need to enhance the defenses against such attacks. The exploitation attempts are relevant mostly to the public-facing applications, like SSH and web servers. Moreover, credential-based attacks, such as SSH brute force attempts, indicate attack vectors executed exploiting the authentication weaknesses, highlighting the importance of robust protocols and password policies. Finally, the behavior of the cyberattacks revealed TTPs relevant to privilege escalation and defense evasion tactics, including sudo executions among others.

4 CONCLUSIONS

In this research paper, we presented the development and architecture of the AI4SIM framework, responsible for simulating advanced AI-powered cyberattacks. Towards this direction, Shennina was validated and the techniques it utilized were extracted as part of the research effort. This process involved assessing its effectiveness in generating AI-powered attacks for simulation purposes within the AI4SIM framework. The comprehensive analysis of the Shenina cybersecurity framework in alignment with the MITRE ATT&CK framework was also presented. Through the examination of the coverage across various MITRE tactics and techniques, this research provides information on the effectiveness and behaviour of the AI-powered offensive tactics that are utilized. By collecting data and extracting information on the observed instance frequencies, the research provided the attack distribution of the Shennina. As a conclusion it should be noted that the approach provided interesting datasets, but the attack vectors are still rather limited, specifically targeting mainly SSH, and HTTP services.

An important aspect deriving from this research was the exploration and alignment of Shennina and in general the signatures generated with the MITRE ATT&CK framework, offering a granular understanding of the capabilities and limitations. By quantifying the coverage and effectiveness of Shenina rules across different tactics and techniques, this research has contributed to the advancement of knowledge in cybersecurity defense strategies. Furthermore, the research paper has highlighted the importance of regular testing and refinement of detection rules based on MITRE TTPs, emphasizing the need for continuous improvement and adaptation in the rapidly evolving threat landscape.

Potential future avenues include the continuous development of the AI4SIM framework to incorporate additional AI techniques and attack scenarios, reflecting the evolving nature of cyber threats. Additionally, an ongoing evaluation and refinement of the effectiveness of the framework to configure and customize the AI cyberattacks will be developed. Furthermore, efforts will be conducted on data collection in order to make them more accessible and usable for researchers.

ACKNOWLEDGMENTS

This work has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101070450 (AI4CYBER).

REFERENCES

[1] Abdul Basit Ajmal, Munam Ali Shah, Carsten Maple, Muhammad Nabeel Asghar, and Saif Ul Islam. 2021. Offensive security: Towards proactive threat hunting via adversary emulation. *IEEE Access* 9 (2021), 126023–126033.

[2] Mariam Alhamed and MM Hafizur Rahman. 2023. A Systematic Literature Review on Penetration Testing in Networks: Future Research Directions. *Applied Sciences* 13, 12 (2023), 6986.

[3] Esra Abdullatif Altulaihan, Abrar Alismail, and Mounir Frikha. 2023. A survey on web application penetration testing. *Electronics* 12, 5 (2023), 1229.

[4] Anna L Buczak and Erhan Guven. 2015. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials* 18, 2 (2015), 1153–1176.

[5] Alessandro Confido, Evridiki V Ntagiou, and Marcus Wallum. 2022. Reinforcing penetration testing using ai. In *2022 IEEE Aerospace Conference (AERO)*. IEEE, 1–15.

[6] Phan The Duy, Nghi Hoang Khoa, Anh Gia-Tuan Nguyen, Van-Hau Pham, et al. 2021. DIGFuPAS: Deceive IDS with GAN and function-preserving on adversarial samples in SDN-enabled networks. *Computers & Security* 109 (2021), 102367.

[7] Lothar Fritsch, Aws Jaber, and Anis Yazidi. 2022. An overview of artificial intelligence used in malware. In *Symposium of the Norwegian AI Society*. Springer, 41–51.

[8] Steven Furnell, Pete Fischer, and Amanda Finch. 2017. Can't get the staff? The growing need for cyber-security skills. *Computer Fraud & Security* 2017, 2 (2017), 5–10.

[9] Mohamed C Ghanem and Thomas M Chen. 2019. Reinforcement learning for efficient network penetration testing. *Information* 11, 1 (2019), 6.

[10] Andreas Happe and Jürgen Cito. 2023. 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*. 2082–2086.

[11] Zhenguo Hu, Razvan Beuran, and Yasuo Tan. 2020. Automated penetration testing using deep reinforcement learning. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2–10.

[12] Eider Iturbe, Erkuden Rios, Angel Rego, and Nerea Toledo. 2023. Artificial Intelligence for next generation cybersecurity: The AI4CYBER framework. In *Proceedings of the 18th International Conference on Availability, Reliability and Security*. 1–8.

[13] Martin Kaloev and Georgi Krastev. 2021. Experiments focused on exploration in deep reinforcement learning. In *2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*. IEEE, 351–355.

[14] Ryusei Maeda and Mamoru Mimura. 2021. Automating post-exploitation with deep reinforcement learning. *Computers & Security* 100 (2021), 102108.

[15] Dean Richard McKinnel, Tooska Dargahi, Ali Dehghantanha, and Kim-Kwang Raymond Choo. 2019. A systematic literature review and meta-analysis on artificial intelligence in penetration testing and vulnerability assessment. *Computers & Electrical Engineering* 75 (2019), 175–188.

[16] George Nakas, Panagiotis Radoglou-Grammatikis, George Amponis, Thomas Lagkas, Vasileios Argyriou, Sotirios Goudos, and Panagiotis Sarigiannidis. 2023. 5G-Fuzz: An Attack Generator for Fuzzing 5GC, using Generative Adversarial Networks. In *2023 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 347–352.

[17] Vasileios Pantelakis, Panagiotis Bountakas, Aristeidis Farao, and Christos Xenakis. 2023. Adversarial Machine Learning Attacks on Multiclass Classification of IoT Network Traffic. In *Proceedings of the 18th International Conference on Availability, Reliability and Security*. 1–8.

[18] Antonio Paya, Sergio Arroni, Vicente García-Díaz, and Alberto Gómez. 2024. Apollon: A robust defense system against Adversarial Machine Learning attacks in Intrusion Detection Systems. *Computers & Security* 136 (2024), 103546.

[19] Sugandh Shah and Babu M Mehtre. 2015. An overview of vulnerability assessment and penetration testing techniques. *Journal of Computer Virology and Hacking Techniques* 11 (2015), 27–49.

[20] Prashant Vats, Manju Mandot, and Anjana Gosain. 2020. A comprehensive literature review of penetration testing & its applications. In *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*. IEEE, 674–680.

[21] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8 (1992), 279–292.

[22] Muhammad Mudassar Yamin, Mohib Ullah, Habib Ullah, and Basel Katt. 2021. Weaponized AI for cyber attacks. *Journal of Information Security and Applications* 57 (2021), 102722.

[23] Xin Yi, Ekta Walia, and Paul Babyn. 2019. Generative adversarial network in medical imaging: A review. *Medical image analysis* 58 (2019), 101552.